#5
W. Lawson
1/19/01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICANT: Grauch et al.

SERIAL NO.: 09/496,825 GROUP ART UNIT: 2711

FILED: February 1, 2000 EXAMINER: C. Grant

FOR: CUSTOMER USAGE TRACKING SYSTEM

ATTORNEY DOCKET NO.: BS95007/086082

Assistant Commissioner for Patents
Washington, D.C. 20231

DECLARATION OF SCOTT SWIX UNDER 37 CFR § 1.131(a)

Sir:

1. My name is Scott Swix. I am over the age of 21 and I am competent to make the declaration based upon my personal knowledge. I understand that this declaration will be used in the United States Patent and Trademark Office ("Patent Office") in connection with the above-identified pending patent application. I also understand that this declaration is being submitted by the owner of the above-referenced application in order to show that the invention claimed in certain claims of that patent application was conceived and reduced to practice in the United States before the November 7, 1995 filing date of the U.S. Patent No. 5,778,182 to Cathey et al. I understand that the Patent Office contends this patent as relevant to at least Claims 1-7 of the above-referenced application.

2. I am presently a project manager at BellSouth Entertainment, which I understand is affiliated with the BellSouth entity that presently owns the above-referenced application. In

order to prepare this declaration, I have reviewed: my work files relating to the development of the invention, for the above referenced patent claims 1-7 of the application, the Cathey et al. patent, § 715.07 from the Manual of Patent Examining Procedure that describes the issues of conception, diligence, and reduction to practice, and 37 C.F.R. §1.31.

3. I am a co-inventor of claims 1 through 7 of the patent application identified above and inventor of the subject matter described and claimed therein. Based on at least the following facts, I believe that my co-inventors and I conceived the Invention at least as early as August 18, 1995.

(1) I worked on the subject matter of the above-identified application ("the Invention"), in the United States, on behalf of BellSouth Corporation ("BellSouth"). In my employment with BellSouth, I was in charge of project development of the Navigator Software Project and was involved with the development of the Invention and its implementation via and in the Navigator software and the BellSouth Digital Broadcast System (BDBS). Additionally, I held and participated in weekly development meetings regarding the Invention.

(2) At least as early as August 18, 1995, my co-inventors and I at BellSouth had conceived the Invention in the United States.

(3) Exhibit 1 is draft version 1.30, dated August 18, 1995, of the design specifications of the Invention, at least some aspects of which we referred to as the "Clickstream system." A copy of this document was filed by BellSouth's counsel as a

disclosure document with the U.S. Patent office on August 19, 1996. Although the disclosure document was filed on August 19, 1996, the filed document is clearly dated August 18, 1995. The document was drafted by a co-inventor, Edward Rowland Grauch, as indicated by the initials "TG'" which stands for Ted Grauch.

B. Beginning at least as early as August 18, 1995, through December 6, 1996, and based on the following facts, I and others worked diligently toward actually implementing or "reducing to practice" the Invention.

(1) Exhibit 2 comprises photocopies from my lab notebook, which include notes taken during meetings in which my co-inventors, staff, and I discussed the Invention and its integration into the Navigator software. For example, page 1 includes notes taken during a steering committee meeting where we discussed boot issues relating to the Invention; page 2 reveals discussions relating to ROM and RAM requirements for the invention; page 3 discloses discussions regarding proof of concept of the Invention; page 4 admits discussions regarding feasibility testing being completed by August 31; page 5 discloses discussions relating to the Invention; page 6 discloses discussions regarding the Invention's control system on the staging server; page 7 reveals discussions relating to what level of analog data could be expected by year end, from the Invention; page 8 discloses that the Invention was collecting data from the navigator during testing; page 9 reveals discussions relating to NVRAM space for the Invention and the memory map ("mem map") associated with Flash ROM and DRAM to be shared by Clickstream and other applications; page 10 admits

discussions relating to the Invention's system support; and page 11 discloses a meeting that took place on January 3, 1997, where we discussed the fact that the Invention was deployed and running in the field.

(2) Exhibit 3 is a test summary report for the Navigator 3.2 software that comprised one way of implementing the invention. The testing results, obtained from a confidential test performed in the United States, completed on December 1, 1996, proved that the Invention had passed quality assurance tests and was ready for release in the market.

(3) Exhibit 4 is a document showing the product release history, in the United States, of BDBS including the Invention. The Invention was first introduced in the Navigator release on November 21, 1996. The Navigator version then deployed was soon thereafter removed in favor of Navigator version 3.22, which successfully implemented the Invention. This version, 3.22, was released and deployed as part of the trial on December 6, 1996.

C. Exhibits attached hereto are photocopies of original documents.

D. I believe that the attached exhibits show that I and the other inventors had conceived the Invention at least as early as August 15, 1995, and, from that date to December 6, 1996, the exhibits show that I and others diligently worked on actually reducing to practice the subject matter of claims 1-7. One version of the Invention was actually reduced to practice at least as early as December 6, 1996 with the deployment of Navigator version 3.22.

3. As the person signing below, I hereby declare that all statements made herein of my

own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under § 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, or any patent issued thereon.

Dated: June 27, 2000

Name: Scott Swix
Citizenship: United States
Post Office Address:
3775 River Hollow Run
Duluth, GA 30096

KILPATRICK & CODY

Suite 2800
1100 Peachtree Street
Atlanta, Georgia 30309-4530

August 19, 1996

Telephone
404 815-6500

Facsimile
404 815-6555

Direct Dial

404 815-6214

DISCLOSURE DOCUMENT NO.

405240

FILING FEE: $10.00
RETAINED FOR 2 YEARS
THIS IS NOT A PATENT APPLICATION

PTO-1652 (4/96)

Other Offices

1400 First Union Bank Bldg.
P.O.Box 2043
Augusta, Georgia 30903

Suite 800
700 13th Street, N.W.
Washington, D.C. 20005

68 Pall Mall
London SW1Y 5ES

Avenue Louise 65, bte 3
1050 Brussels

CERTIFIED MAIL #Z 012 659 8
RETURN RECEIPT REQUEST

Assistant Commissioner
for Patents
Box DD
Washington, D.C. 20231

Re: Disclosure Document Program -- Clickstream System
Our Ref.: BS116/093485

Dear Sir:

Enclosed for submission with the Disclosure Document Program in accordance with the Manual of Patent Examining Procedure § 1706 are (1) the original of a paper signed by the inventor requesting that the enclosed materials be received for processing under the Disclosure Document Program and (2) our check in the amount of Ten Dollars ($10.00). The Commissioner is authorized to debit Kilpatrick & Cody deposit account no. 11-0855 for any deficiency in the enclosed fee, and a copy of this paper is enclosed.

Please date, stamp and return the enclosed, self-addressed postcard, also, please date stamp and return the enclosed copy of this letter in the self-addressed stamped envelope provided to evidence receipt of the letter and the referenced enclosures.

Sincerely,

Mitchell G. Stockwell, Reg. No. 39,389

MGS/sf
Enclosures

Entered Computer 9/20/96
P053:5

267017.1

The date stamp on the card acknowledges receipt of the enclosed:

Applicant(s): John Batton, Ted Grauch and Tom Danner
Title: Clickstream System

Disclosure Document
Filing Date: August 19, 1996
Docket No.: BS116/093485

Executed Disclosure Document Cover Letter to Assistant Commissioner for Patents;
Product Description Sheet;
Executed Signature Page with Two Witnesses;
Check No. 102028 in the amount of $10.00 for disclosure document recordation;
Duplicate of Mitchell G. Stockwell, Esq. Letter Requesting Recordation

KILPATRICK & CODY

Suite 2800
1100 Peachtree Street
Atlanta, Georgia 30309-4530

Telephone
404 815-6500

Facsimile
404 815-6555

Direct Dial
404 815-6214

Other Offices

1400 First Union Bank Bldg.
P.O.Box 2043
Augusta, Georgia 30903

Suite 800
700 13th Street, N.W.
Washington, D.C. 20005

68 Pall Mall
London SW1Y 5ES

Avenue Louise 65, bte 3
1050 Brussels

August 19, 1996

CERTIFIED MAIL #Z 012 659 845
RETURN RECEIPT REQUESTED

Assistant Commissioner
for Patents
Box DD
Washington, D.C. 20231

Re: Disclosure Document Program -- Clickstream System
Our Ref.: BS116/093485

Dear Sir:

Enclosed for submission with the Disclosure Document Program in accordance with the Manual of Patent Examining Procedure § 1706 are (1) the original of a paper signed by the inventor, requesting that the enclosed materials be received for processing under the Disclosure Document Program; and (2) our check in the amount of Ten Dollars ($10.00). The Commissioner is authorized to debit Kilpatrick & Cody deposit account no. 11-0855 for any deficiency in the enclosed fee, and a copy of this paper is enclosed.

Please date, stamp and return the enclosed, self-addressed postcard, also, please date, stamp and return the enclosed copy of this letter in the self-addressed stamped envelope provided to evidence receipt of the letter and the referenced enclosures.

Sincerely,

Mitchell G. Stockwell, Reg. No. 39,389

MGS/sf
Enclosures

Entered Computer 8/22/96
P05395

267017.1

MGS

Z 012 659 845

Receipt for Certified Mail 08348S-

No Insurance Coverage Provided
Do not use for International Mail
(See Reverse)

UNITED STATES POSTAL SERVICE

Sent to Asst. Comm. for Patents	
Street and No	
P.O., State and ZIP Code Box DD, Washington DC 20231	
Postage	$
Certified Fee	
Special Delivery Fee	
Restricted Delivery Fee	
Return Receipt Showing to Whom & Date Delivered	
Return Receipt Showing to Whom, Date, and Addressee's Address	
TOTAL Postage & Fees	$
Postmark or Date	

CENTRAL CITY FINANCIAL
JUL 9 1996
22305

PS Form 3800, March 1993

SENDER:

- Complete items 1 and/or 2 for additional services.
- Complete items 3, and 4a & b.
- Print your name and address on the reverse of this form so that we can return this card to you.
- Attach this form to the front of the mailpiece, or on the back if space does not permit.
- Write "Return Receipt Requested" on the mailpiece below the article number.
- The Return Receipt will show to whom the article was delivered and the date delivered.

I also wish to receive the following services (for an extra fee):

1. ☐ Addressee's Address
2. ☐ Restricted Delivery

Consult postmaster for fee.

3. Article Addressed to:

Assistant Commissioner for
Patents
Box DD
Washington, D.C. 20231

RECEIVED AUG 22 1996 PTO MAIL CENTER CLERK #104

4a. Article Number
Z 012 659 845

4b. Service Type
☐ Registered ☐ Insured
☒ Certified ☐ COD
☐ Express Mail ☐ Return Receipt for Merchandise

7. Date of Delivery

5. Signature (Addressee)
MGS 08/16/96

8. Addressee's Address (Only if requested and fee is paid)

6. Signature (Agent)

PS Form 3811, December 1991 ☆U.S. GPO: 1993—352-714 **DOMESTIC RETURN RECEIPT**

Is your RETURN ADDRESS completed on the reverse side?

Thank you for using Return Receipt Service.

Z 012 659 845

Receipt for Certified Mail

No Insurance Coverage Provided
Do not use for International Mail
(See Reverse)

Sent to Asst. Comm. for Patents	
Street and No	
P.O., State and ZIP Code Box DD, Washington DC 20231	
Postage	$
Certified Fee	
Special Delivery Fee	
Restricted Delivery Fee	
Return Receipt Showing to Whom & Date Delivered	
Return Receipt Showing to Whom, Date, and Addressee's Address	
TOTAL Postage & Fees	$
Postmark or Date	

PS Form 3800, March 1993

Fold at line over top of envelope to the right of the return address

CERTIFIED

Z 012 659 845

MAIL

Is your RETURN ADDRESS completed on the reverse side?

SENDER:
- Complete items 1 and/or 2 for additional services.
- Complete items 3, and 4a & b.
- Print your name and address on the reverse of this form so that we can return this card to you.
- Attach this form to the front of the mailpiece, or on the back if space does not permit.
- Write "Return Receipt Requested" on the mailpiece below the article number.
- The Return Receipt will show to whom the article was delivered and the date delivered.

I also wish to receive the following services (for an extra fee):

1. ☐ Addressee's Address
2. ☐ Restricted Delivery

Consult postmaster for fee.

3. Article Addressed to:

Assistant Commissioner for
Patents
Box DD
Washington, D.C. 20231

4a. Article Number
Z 012 659 845

4b. Service Type
☐ Registered ☐ Insured
☒ Certified ☐ COD
☐ Express Mail ☐ Return Receipt for Merchandise

7. Date of Delivery

5. Signature (Addressee)

6. Signature (Agent)

8. Addressee's Address (Only if requested and fee is paid)

PS Form 3811, December 1991 ☆U.S. GPO: 1993—352-714 **DOMESTIC RETURN RECEIPT**

Thank you for using Return Receipt Service.

ILPATRICK & CODY

WACHOVIA ACCOUNT NO.: 12 659 288

E: 08-02-96 PAYEE: ASST COMMISSIONER FOR PATENTS VENDOR # 18464 CHECK #: 102028

REF.#:	G/L #	INVOICE #	INVOICE DATE	DESCRIPTION	AMT.PAID
4423	20999 00000000	25	08-02-96	093485 Filing of Disclosure Documents.	10.00
				TOTAL	10.00

KILPATRICK & CODY
OPERATING ACCOUNT
1100 Peachtree Street
Atlanta, Georgia 30309-4530

DATE
08-02-96

WACHOVIA
Wachovia Bank of Georgia, N.A.
Atlanta, GA 30383

64-1
610

NO. 102028

$*******10.00**

AY **TEN AND 00/100 Dollars**

VOID AFTER 180 DAYS
TWO SIGNATURES REQUIRED OVER $2,000

O
HE
RDER
F **ASST COMMISSIONER FOR PATENTS**

⑈102028⑈ ⑆061000010⑆ 12 659 288⑈

The date stamp on the card acknowledges receipt of the enclosed:

Applicant(s): John Batton, Ted Grauch and Tom Danner
Title: Clickstream System

Disclosure Document
Filing Date: August 19, 1996
Docket No.: BS116/093485

Executed Disclosure Document Cover Letter to Assistant Commissioner for Patents;
Product Description Sheet;
Executed Signature Page with Two Witnesses;
Check No. 102028 in the amount of $10.00 for disclosure document recordation;
Duplicate of Mitchell G. Stockwell, Esq. Letter Requesting Recordation

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Assistant Commissioner for Patents
Box DD
Washington, D.C. 20231

DISCLOSURE DOCUMENT

Sir:

The undersigned, being the inventors of the disclosed invention, request that the enclosed papers be accepted under the Disclosure Document Program, and that they be preserved for a period of two years.

Respectfully submitted,

John Batten

Ted Grauch

Tom Danner

Date: 7/8/96

Enclosures

Clickstream System

Design Specification

for the BellSouth Interactive Video Services Network Trial

DRAFT

*BELL*SOUTH Interactive Media Services

This document specifies software designs to enable the journaling of subscriber viewing habits for statistical analysis.

For input regarding this document contact: Ted Grauch (404)-392-5664

Revision History:

1.00Draft	1/20/95	5:00pmEST	TG	First release
1.01Draft	1/24/95	9:00amEST	TG	Corrections
1.10Draft	2/16/95	1:00pmEST	TG	Separated CONDO from OPTIMARK functions, updated data fields to match evolving CONDO standard, added timed upload feature.
1.11Draft	3/15/95	1:00pmEST	TG	Updated CONDO data fields, Changed upload flow for single session STB.
1.20Draft	4/21/95	5:00pmEST	TG	Updated Name, Added overview of other system support, Full functional description of STB process, Application Identifiers, Time and Metadata Formats, and Metadata merge functions.
1.30Draft	8/18/95	5:15pmEST	TG	Refined Clickstream processor information, Journalling Technique, Data Formats, Defined 3rd Party App API, Changed CONDO references to I^3, Refined data merge spec., Expanded fault tolerance handling.

CONTENTS
Clickstream System Specification

1. Requirements of the Clickstream System

The Clickstream System is a software system residing on several hardware platforms within the BellSouth Interactive Video Services Network (IVSN). The system is being designed for the BellSouth Residential Broadband Trial which will take place in the city of Chamblee, Georgia from May, 1995 through May, 1997. The purpose of the system is three fold:

1) To provide the capability for the collection of knowledge about the subscriber usage of the BellSouth System and Services. This is called Clickstream information. This knowledge will allow analysis of data to generate detailed viewing habits, programming ratings, and detailed demographics of the subscriber population.
2) To provide a method (or methods) to merge content identifier Metadata with the Clickstream information.
3) To provide a storage and organization database system to manage and analyze the Clickstream and content data, and to provide an upload facility to 3rd party analysis entities, such as NCM I^3/Optimark. This system has been commonly referred to as the MarKeting and Information System (MKIS).

These 3 main functions of the Clickstream system are tailored to support the analysis of subscriber Interactive Video Services Network (IVSN) usage. Some analysis capability will be supported by the MKIS system, but main analysis function will be provided off-line by a third party (NCM/BULL/Arbitron) system referred to as I^3/ Optimark. For a more detailed account of data analysis function, refer to NCM/BULL/Arbitron documentation.

2. Clickstream Hardware/Software Entities

The software subsystems to support the Clickstream collection, Metadata merging, and data storage run on several hardware platforms as outlined in figure 1. The subsystems communicate with each other in various ways to transmit Clickstream information, data error detection schemes, and data acknowledgments. Only the flow of Clickstream pertinent data and Metadata are displayed in figure 1. The software subsystems are divided functionally within each platform to aid in more parallel design, development and testing efforts.

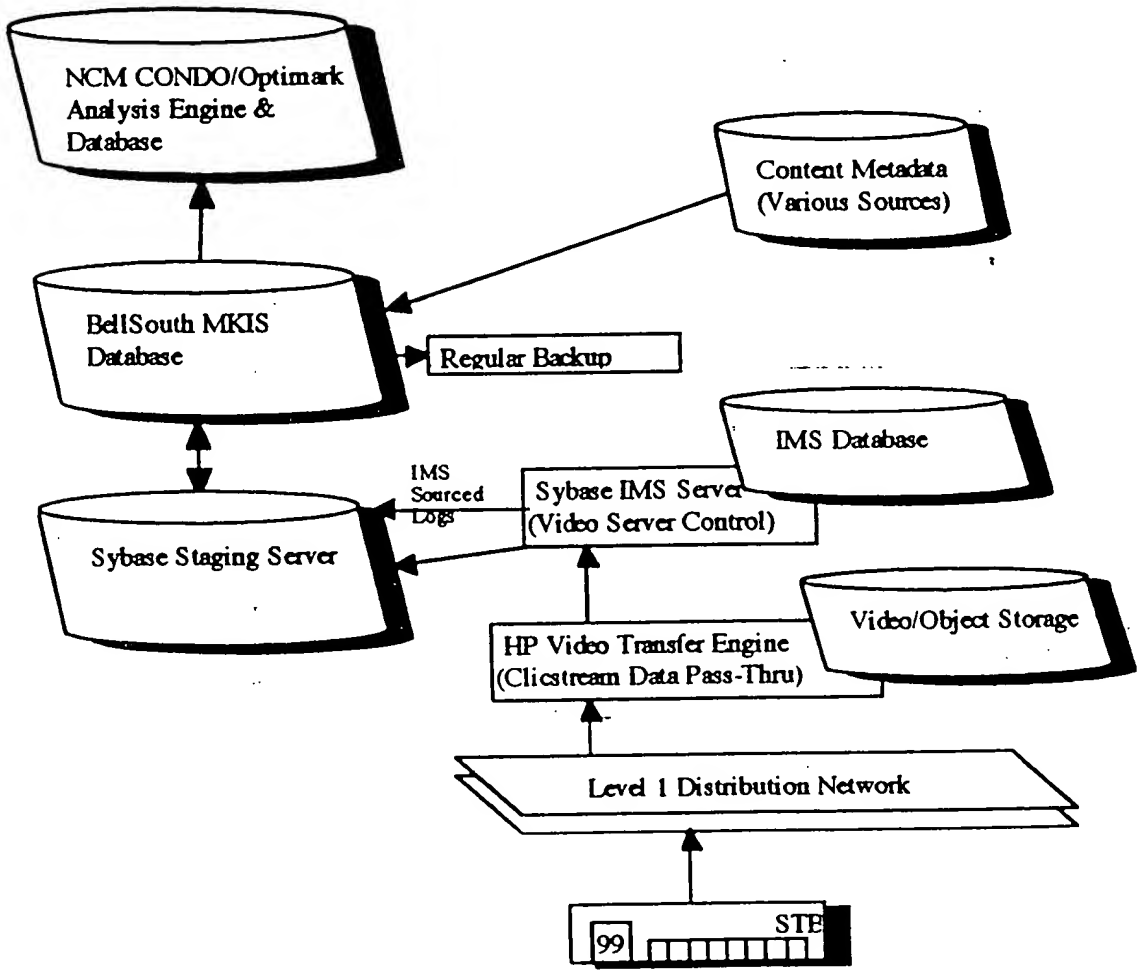


Figure 1: Clickstream System Data Flow

Note: Arrows represent flow of Clickstream data only

2.1. *Software Subsystems*

The following is an overview of expected software subsystems, in-depth functionality will be covered in following sections.

2.1.1. STB Platform

There are several parts to the STB based Clickstream Processor code:

1) Clickstream Kernel
2) Double Buffer of Clickstream Events
3) Clickstream Upload Handler
4) Clickstream Message Receiver, Upload Controller
5) Clickstream Event API

The STB based Clickstream Processor will reside in, and be executed from DRAM. It will be downloaded to the STB during the Level 2 boot process, and will be designated as a "Resident" application by the PowerTV Operating System (It will remain memory resident over soft powerdown STB states).
The Clickstream Kernel will buffer Clickstream Events handed to it by applications.
The Clickstream Message Receiver will accept control messages over the ESF Pass-through data link to control the uploading of information over the reverse path network and will store the payload of these messages in appropriate and available memory (NVM when possible). It will also accept the messages sent to it to acknowledge the receipt of a number of Clickstream Data Packets.
The uploading of data will occur on a scheduled basis over every 24 hour time period. Clickstreams will be uploaded over the system messaging or "L1 Pass-Thru" protocol. This scheduling function will allow the distribution of uploads evenly over 24 hour periods to minimize network impact. Uploads may also be suspended every day during high network usage times (prime time, etc.).

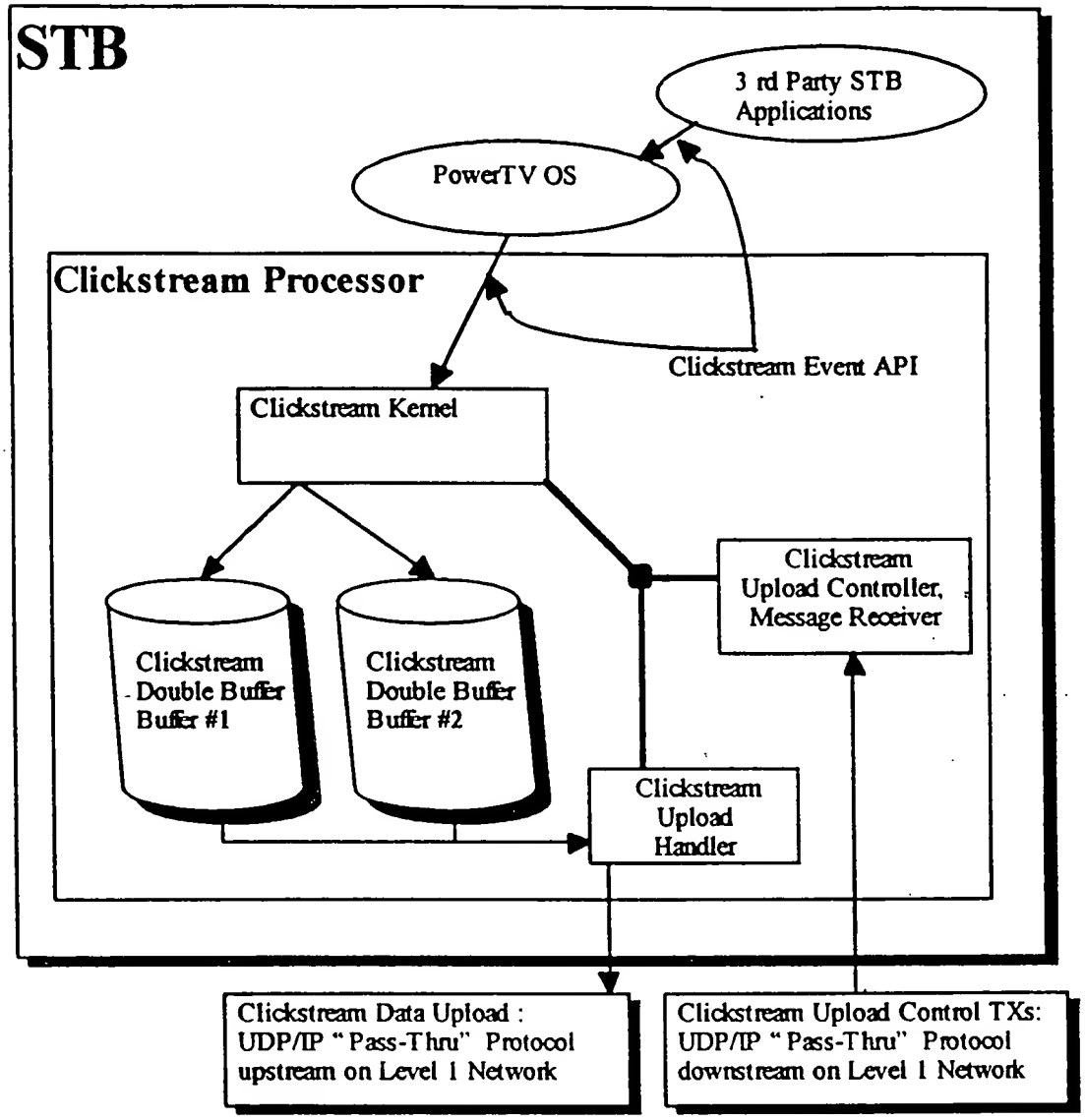


Figure 2: STB Clickstream Journalling Flow

2.1.2. HP VTE Platform

The video server simply acts as a pass-thru device for both STB sourced Clickstream Uploads, and for data ACKs returned to the STB. There is a communications router inside the VTE which will re-direct IP traffic to the appropriate destination.

2.1.3. Sybase IMS Platform

The IMS platform will have two functions pertaining to the Clickstream system. One will be to act as a pass-thru device for STB sourced Clickstream information going to the Staging Server, and as a pass-thru device for Staging Server data acknowledgments to the STB. This communication will be handled through the Sybase Open Client/OpenServer Software and will take the form of Remote Procedure Calls (RPCs) executed on the IMS. The other function will be to replicate IMS log information to the Staging Server for inclusion in the Metadata which is further replicated to the MKIS.

2.1.4. Sybase Staging Server Platform

The Sybase Staging Server will have four main functions.

1) It will act as an endpoint in the Clickstream upload process. This is handled by a process and flat file persistant storage called "Event Capture". It will use the RPCs made from the IMS server to pass Clickstream information over, and formulate and send data acknowledgments to the STB.

2) It will handle replicated IMS data logs destined for the MKIS system.

3) It will have a decompression, parsing, and data merging mechanism which will separate the Clickstream data into appropriate relational database entities as outlined in the MKIS specification.

4) It will act as a persistent storage device from which MKIS database information can be replicated.

2.1.5. MKIS Platform

The MKIS system will serve several functions within the Clickstream system. The particular design of this database and analysis engine is covered extensively in the document **"IT Architecture Design Ver. 1.1"**. A top level look at this system reveals 4 main functions.

1) Mass data repository and relational database structure for:
Clickstream, subscriber, demographics, billing , advertisement and content information

2) To enable regular data uploads to third party analysis engines, and to be able to restrict access to sensitive data fields.

3) To perform "native" data analysis functions and report generations to different specifications.

3. STB Clickstream Processor

3.1. Overview

The Clickstream processor is structured to store Clickstream event records in one of two Clickstream double buffers. The Clickstream buffer sizes are staticly provisioned by the PowerTV OS as defined in the code download process. The buffer size will be optimized during the test and integration phase of the project. The possibility has not been ruled out to addressably download the size of the Clickstream STB double buffer. The size of any particular Clickstream event record is variable to handle the different journaling needs of different applications. An event record is the individual logging of a subscriber input to the STB. The generation of these events fill one of the two Clickstream buffers. When a buffer is filled, or an upload timer event has expired, the Clickstream processor is flagged to initiate an upload process with the Staging Server based Event Capture application, and the buffer becomes locked. Subsequent Clickstream records are stored to the second Clickstream buffer.

The upload process consists of several steps. The buffer to be uploaded through the network is locked. No more Event Records can be stored to this buffer. The buffer is then compressed. The PowerTV OS will support the LZW compression utilities. These will be employed to compress the buffer. Fifty percent compression is acheivable with this algorithm on random binary data. Clickstream data roughly fits into this category. The resultant compressed data is then divided into transmission "transactions" or "packets" and data is placed into the packet headers to indicate packet ID, IP destination address, etc. CRC error detection will either be placed on each packet by the Clickstream Processor (application level), or by the UDP drivers in the OS. This issue is currently TBD. Each data packet has UDP/IP protocol around a L1 pass-thru header. The final bit-format level specifications have not yet been received from PowerTV, but will be included in this specification on their receipt.

The Clickstream data packets are uploaded over the Slotted-ALOHA data transmitter out of the STB. The Staging Server based "Event Capture" application will receive these records and transmit addressable responses to the STB in the form of data acknowledgments. The frequency and periodicity of data acknowledgements will be determined during the implementation stage of the project. Correct implementation of application level data acknowlegements require empirical knowlege of network bit error rates, netowork packet error rates, and causes of these errors in transmission.

If the second buffer becomes completely full and the upload process has not completed, then a buffer overrun condition will occur. The last record in each Clickstream buffer is reserved for a buffer overrun record with a timestamp, this essentially tells the Staging Server resident application that STB clicks were missed starting at a particular time. This error condition can then be corrected by increasing the buffer sizes.

(NOTE: Receipt of a buffer overrun record will tell us that buffer sizes have not been set appropriately, and we will have to re-set buffer sizes, re-compile our code and release it to the system as an update. We may also implement an addressable parameter setting the Clickstream buffer sizes dynamicly. (There are reasons both pro and con to do this). Appropriate buffer sizes should be determinable within the first month of integration testing.)

3.2. Clickstream Kernel

The Clickstream code will be downloaded to the STB during the Level 2 (L2) boot sequence as outlined in the BellSouth/Sybase document "ITV Process Flows Rev. 1.00". It will be bundled along with all Binary Large OBjects (BLOBs) intended to be memory resident , and downloaded at initial AC power on following the Level 1 (L1) bootterm . Bundling of services takes place within the IMS server and takes the form of a "list of services" which is the first item downloaded to the STB on L2 Boot. There is a memory allocation and a prioritization table associated with each of these applications downloaded into DRAM. They are contained in what PowerTV terms a "Dispatch Table" included with each downloaded application. For more information on lists of services and the code download process, see Sybase IMS documentation, and the PowerTV document "Loader Specification", and "Application Download Procedure".

3.3. Clickstream Buffering

The database is allocated into a double buffer (two blocks of memory). Each block is a contiguous free area of DRAM which is set aside for this application use only. The buffer sizes should not need to be very large (10K-15K?), so allocation of contiguous blocks should not be a problem. Utilization of more advanced database techniques such as linked lists or record pointers should not be necessary in this type of application and would only increase the application footprint and complexity. The database buffers should be allocated to allow at least 4 to 8 hours of peak channel "surfing" between uploads. This will lead to provisioning the actual size of the data buffers after we get some empirical results on subscriber usage. This should keep system bandwidth needs within reason. Data compression techniques will be employed to further reduce the transmission bandwidth needs.

The Clickstream event records consist of data specified by Arbitron/NCM in the document "Database Design Specification NCM I [3]", plus additional fields used on an application specific basis to allow for flexible documentation of subscriber usage of the interactive system. This data is collected from two sources by the STB when an event is registered. The application generating the Clickstream event hands off data regarding the application state and the subscriber UI which caused the event. The application will also hand over information obtained from the OS such as timestamp and STB id.

3.4. Clickstream Event Format

3.4.1. General Header Format for Events

The general event format is tailored to be as flexible as possible. Each application running on the STB will interface to the Clickstream routines to send just the data type and format that it needs to journal.

The general information sent by each application will uniquely identify the time of event generation to the second, the application which generated the event and the number of bytes to follow which constitute the information the application wishes to journal.

(**Developer Note:** The Application ID sent in the API call to the Clickstream Processor and stored in the Buffer as shown below is **NOT** the Application ID given to your application by the Operating System during initialization, but the Application ID listed in section 3.4.3 of this document. This identifier is unique and static and will enable backend systems to parse the events journaled by your application. The application ID given to you by the operating system at application initialization is dynamic, and will change every time your application is launched. If there is no Application ID listed for your application in Table 4 of section 3.4.3, please contact current contact person listed on the cover of this document to have one assigned to you.)

Syntax	Size/Format
Clickstream_Buffer ()	
{	
Buffer Header Record {	
Transaction Code	ui8
Clickstream Version Number	ui8
Timestamp	ui8 * 6
Number of Bytes	ui8
STB MAC Address Most Significant	ui16
STB MAC Address Least Significant	ui32
Compression Type	ui8
}	
Event Record	
for (i=0; i<N; i++) {	
Timestamp	ui8 * 6
BIMs Assigned Application ID	ui16
Number Bytes to Follow (length)	ui8
Application Specific Data see section 3.4.4 for specific formats used by each application.	ui8 * length
}	
Clickstream_Trailer Record {	
Timestamp	ui8 * 6
BIMs Assigned Application ID	ui16
Number Bytes to Follow (length)	ui8
Upload Status Code	ui8
}	
}	

N = Number of Event Records able to be stored in this Clickstream_Buffer.

Figure 3: Clickstream Buffer Format

There are two buffers such that one may be "frozen" during the upload process, and the second buffer can then be employed to continue journaling event records. Both buffers are of identical format. This format is shown in Figure 2 above. The notation used in Figure 2 is ISO standard for description of data structures.

The Clickstream Buffers are formatted specifically to ease in their transmission back through the distribution network to the Staging Server "Event Capture" data repository. The first section of entries comprise a header record which will indicate the time the buffer was first opened, the number of bytes in the buffer, will define the originating STB by MAC Address, defines the version of the Clickstream Kernel which generated the buffer, and specifies the type of data compression used on the following data (if any). This first record is of fixed length and is never compressed. All bytes following "Compression Type" are possibly compressed to save in transmission bandwidth.

The Clickstream_Trailer record appears at the end of a Clickstream Buffer and usually denotes an error condition. If the upload capabilities of the system did not allow the clearing of data from the STB Clickstream Buffers. This record distinguishes itself by the Application ID pointing to a "Missed Records" type so that queries can be performed on this data in one of our backend systems. The Upload Code is then used to identify the possible cause for the error condition. These error codes are outlined in section 3.4.3.

3.4.2. Upload Codes

Upload Codes in the general event format are meant as an indicator of what state the Clickstream application was in when a buffer overflow occurred. This code indicates what stage in the upload process the Clickstream kernel was in at the time of overflow . This will help us determine the cause of buffer overflows, and the appropriate way to solve these problems during testing and integration.

Upload Codes	
0x00	Not Used
0x01	Upload in Progress
0x02	Upload Completed, No Acknowledgement Received
0x03	Upload Completed, Partial Acknowledgements Received
0x04	No Upload Attempted

Figure 4: Upload State Codes

3.4.3. Application Unique Identifiers

Each application with the ability to operate on a BellSouth Interactive STB will be assigned a 16 bit unique identifier. In addition, each application will have the option of posting an 8 bit Application State identifier within it's application specific data. This document, and further revisions, will act as the authority for the assignment of these identifiers. For the applications known at this time, the following table specifies these "APP IDs".

Application Identifiers	
0x0000	Operating System
0x0001-F	Operating System Sub-Systems (TBD)
0x0010	Application Manager
0x0011	Cable Television Application
0x0012	Clickstream Set-Top Box Kernel
0x0100	Prevue Interactive Services - EPG System
0x0101	Digital Pictures - Interactive Game
0x0110-F	Viacom - MTV / Showtime, etc.
0x1000	Interplay Written Applications General ID (TBD)
0x1001	Interplay Runtime Engine
0x1002	Interplay Navigator
0x1003	Interplay VOD
0x1004	Interplay NVOD
0x1005	Interplay TownGuide
0x1100	The Weather Channel, Weather On-Demand
0x1101	Worldspan - Travel On-Demand
0x1102	Lightspan - Educational Interactive Application
0x1103	
0xFFFF	Missed Events Record

Figure 5: Application Unique IDs

3.4.4. Application Specific Event Formats

Developer Note: The Clickstream formats proposed in this section are close to what we currently believe will be the final data formats. As we get a better understanding of all of the analysis needs, these data formats may change.

3.4.4.1. Operating System

The operating system will have the ability to make the same call to the Clickstream system as any other code residing on the STB. It may use this facility to log errors, for testing purposes, to track code usage, or for any other reasons. The OS tie-in to the Clickstream system will be predicated by PowerTV's interest in being involved, and is not seen as likely in the short term.

3.4.4.2. Application Manager

Application Manager is a portion of the "Level 1 Client" as defined in PowerTV operating system documentation. This code will be responsible for managing the execution of all applications on the STB, handling application switching, removing applications which are mis-behaving, and the like. BellSouth will be interested in logging of all application switching activities. A data format has been established to this end. Implementation of Application Manager posting Clickstream Events will depend on schedules and interest level of PowerTV.

Syntax	Size/Format
...ion_Manager: Application_Specific_Data ()	
{	
Event Record	
for (i=0; i<N; i++) {	
Event ID: See Global Event ID table for Syntax	ui16
Suspend Application ID	ui16
Initialize Application ID	ui16
Application Error Code: See below for Syntax	ui8
}	
}	

Error Code Syntax	Data
Application_Manager: Application_Error_Code ()	ui8
{	
Do not use (No Error)	0x00
Application termination reason unknown	0x01
Missed Watchdog Timer (Application Keep-Alive)	0x02
Unexpected Session Teardown	0x03
Memory Error	0x04
Application will not download	0x05
Communication Error	0x06
Network Error	0x07
}	

Figure 6: Application Manager Event Data Format

3.4.4.3.Cable TV Application

The Cable Application is the second half of what PowerTV calls the "Level 1 Client". It is responsible for tuning both analog and digital broadcast services.

Syntax	Size/Format
Cable_Application : Application_Specific_Data ()	
{	
Event Record	
for (i=0; i<N; i++) {	
Event ID: See Global Event ID table for Syntax	ui16
Channel ID : See Broadcast Channel ID table for Syntax	ui16
}	
}	

Figure 7: Cable Application Event Data Format

The Channel ID 16 bit field is a unique identifier for broadcast networks. In analysis of data, the fact that channel 6 was watched more than channel 7 has little or no meaning to BellSouth. When the network associated with the channel is used, the data becomes both clearer and more valuable. To illustrate, the fact that "TBS" was watched more frequently than "MTV" would be valuable information, and is more in line with our end requirements. This encoding method also allows the data to stay consistent across channel lineup changes that will take place during the trial, or as the system grows past technical trial this encoding scheme will allow Clickstream data from various headends in locations all over the southeast to generate consistent data no matter what their individual channel lineups may be. A list of all major broadcast networks in the US appears in the BellSouth John Stefanik authored document "Residential Broadband Applications Guide V1.10". The encoding scheme for these broadcast providers is defined in this document in the Metadata formats section. The Event ID as listed above defines the action which occurred on that channel at that time and is also defined in the Metadata section of this document.

3.4.4.4.Clickstream STB Kernel

The Clickstream Kernel may generate Clickstream events itself to log errors in Clickstream collection and/or transmission. We will have a better handle on the types of errors or faults we would like to capture during our test and integration phase. This information will be folded into this design document on an as-needed basis.

3.4.4.5. Navigator

The Navigator is the interactive menuing system which enables viewers to select from our many interactive services. This is a BellSouth initiative and we will be collecting usage data to analyze how easily viewers are able to access programming they are after.

Syntax	Size/Format
Navigator : Application_Specific_Data ()	
{	
Event Record	
for (i=0; i<N; i++) {	
Event ID: See Global Event ID table for Syntax	ui16
Application State ID: See table below for Syntax	ui8
for (i=0; i<N1; i++) {	
Character String: ASCII, Variable Length	b8 * N1
}	
}	
}	

State_ID	Data
Navigator: Application_State_ID ()	**ui8**
{	
Fly-Thru	0x00
Main Menu	0x01
Information (Help) Screen or Video	0x02
Movies Sub-Menu	0x03
Movie Categories Sub-Menu	0x04
List of Movies Sub-Menu	0x05
Movie Info Screen	0x06
Movie Buy State	0x07
}	

Figure 8: Navigator Buffer Format

3.4.4.6. VOD

Video On Demand will be launched from the Navigator and will be a BellSouth service. Clickstream capturing will be interested in the amount of pausing, Fast-forwarding and Rewinding people do, as well as what is being viewed at a particular point in time. Some of this will change as we get a better idea of exactly what events are logged by the IMS server. We will try not to duplicate effort wherever possible. If it is determined that the IMS Event Log is capturing this level of information, then the Clickstream Journalling of this information is redundant and will not be implemented.

Syntax	Size/Format
Video On Demand (VOD) : Application_Specific_Data ()	
{	
Event Record	
for (i=0; i<N; i++) {	
Event ID: See Global Event ID table for Syntax	ui16
Application State ID: See table below for Syntax	ui8
}	
}	

State_ID	Data
VOD: Application_State_ID ()	**ui8**
{	
Playing	0x00
Paused	0x01
Fast Forward	0x02
Rewind	0x03
Info (Help) Video or Screen Played	0x04
reserved	0x05
reserved	0x06
reserved	0x07
}	

Figure 9: VOD Buffer Format

3.4.4.7.NVOD

This application is also a BellSouth service. The following tables outline expected application specific event formats, and application states possible within the application. The NVOD application will be active without an active Client/Server connection to the IMS. Clickstream System will be the oly mechanism for BellSouth to glean usage information on this application.

Syntax	Size/Format
Near Video On Demand (NVOD) : App_Specific_Data ()	
{	
Event Record	
for (i=0; i<N; i++) {	
Event ID: See Global Event ID table for Syntax	ui16
Application State ID: See table below for Syntax	ui8
}	
}	

State_ID	Data
NVOD: Application_State_ID ()	ui8
{	
Playing	0x00
Incremental Pause	0x01
Incremental Skip Forward	0x02
Incremental Skip Backward (Rewind)	0x03
Info (Help) Video or Screen Played	0x04
reserved	0x05
reserved	0x06
reserved	0x07
}	

Figure 10: NVOD Buffer Format

3.4.4.8.EPG

Prevue Guide Interactive Services are expected as the Electronic Program Guide (EPG) provider for the BellSouth Interactive Trial.

Syntax	Size/Format
Electronic Program Guide (EPG) : App_Specific_Data ()	
{	
Event Record	
for (i=0; i<N; i++) {	
Event ID: See Global Event ID table for Syntax	ui16
Application State ID: See table below for Syntax	ui8
reserved	ui8
}	
}	

State_ID	Data
EPG: Application_State_ID ()	ui8
{	
Initial Display Screen	0x00
Look Ahead Display 4 hour	0x01
Look Ahead Display 8 hour	0x02
Look Ahead Display 12 hour	0x03
Look Ahead Display 16hour	0x04
Look Ahead Display 20 hour	0x05
Look Ahead Display 24 hour	0x06
reserved	0x07
}	

Figure 11: EPG Buffer Format

3.4.4.9. Home Shopping

Expected data formats will be developed following the successful deployment of initial offerings of interactive applications in 1995.

3.4.4.10. Other

Expected data formats for a large number of interactive applications will be developed following the successful deployment of initial offerings of interactive applications in 1995.

3.5. *Applications Programming Interface (API) to Clickstream*

To interface with various third party applications, the Clickstream Processor has utilized the cooperative nature of the PowerTV operating system which allows for internal point to point messaging. The PowerTV operating system uses the concept of EVENT very broadly to handle everything from thread usage, semaphores, system thread contention, timer handlers, exception handling, etc. Using the EVENT system, BellSouth has defined a unique Operating System Event Type:

(kEt_clickStream), which will be used for inter-process communication with the Clickstream Kernel. This allows us to get away from the alternate design of having to support Dynamic Link Libraries (DLLs) for all interested applications, and the support and complication it would have introduced into the system.

Developer Note: Make sure that the difference between the "Operating System concept of Event" and the generation of a "Clickstream event" remain somewhat clear to you, they can be easy to confuse one for the other.

The Clickstream event, or Clickstream Journalled event is a data set that identifies subscriber action on the ITV system at a particular point in time. e.g. "Channel was incremented once from channel 5 at 5:12PM on Tuesday." The PowerTV Operating System Event system and PowerTV Event data structure are generic ways the OS gets its job done. The system and the data structure are employed to handle a myriad of OS responsibilities. We are simply using this Event system as a way to get our Clickstreams from point A to point B within the Set-Top Box.

3.5.1. PowerTV OS Events:

Running under the PowerTV OS, an application can generate Events, receive Events, filter reception of Events, etc. The Event itself consists of 5 parameters:

PowerTV General Event Data Structure			Size/Format
Event ()			
{			
Code	:	Defines the Event Type, Event source, what type of device generated the Event, and a general 8bit field for Event data.	ui32
X	:	Application specific data field.	ui32
Y	:	Application specific data field.	ui32
Z	:	Application specific data field.	ui32
T	:	System time when Event was generated. (this is from a 64 bit PowerPC register and has no link to real world time, it is used for processing the Event.)	ui64
}			

Figure 12 : OS Event Data Structure

An application will start receiving Events when it uses PowerTV APIs to "register interest" in a particular, or a set of particular Events. An application can start generating Events at any time by simply using the appropriate APIs. The Code field of the Event is what PTV uses your mask to check against any Events that are generated. A quick look at the way the Code field has been structured (above) should demonstrate that there are several ways the Code can be used to filter and route Events. When the App registers Event interest it defines an Event mask in terms of the Code field. The OS then uses this mask along with all others it has received to decide where an Event should be routed, and in which Event Queues it should be placed. When Apps register interest they also specify a DRAM memory location called an Event Queue which acts analagous to a data mailbox. If your application asks for a lot of Event types, (e.g. "Give me all Events generated by all external devices") you will receive alot of "mail" in your Event Queue memory locations, if you ask for only certain Events (e.g. "Give me an Event only when the Power button is pressed") , you will receive just sporadic Events in your "mailbox".

3.5.2. Clickstream Event type: "kEt_clickStream":

BellSouth has defined a unique Operating System Event Type for use in the BellSouth system, and has done so with PowerTV approval. Today this Event Type is defined as a "#define" in our application source, but will appear in the PTVTYPES.h file which PowerTV releases as part of their standard Operating System release in the near future. (This #define will only appear in releases of PowerTV for BellSouth use and 3rd party application developers for BellSouth.)

Clickstream Kernel is initialized following Level 2 boot by the PTV OS Application Manager. During it's initialization process it registers interest with the PowerTV OS in any OS Event of the "kEt_clickStream" type. It is the only application which registers interest in this Event type. When an application with focus (or otherwise) reaches a point in code where an action worthy of journalling has occured, it places the contents of the Clickstream Journalled event into a known area of DRAM. It then launches a kEt_clickStream type Event. With a kEt_clickStream Event the following syntax is used:

kEt_clickStream Structure	Contents	Size/Format
kEt_clickStream ()		
{		
Code :	0x00008000	ui32
X : Clickstream Application ID	See section 3.4.3 Fig. 5 for ID used by each Application. Fill with leading Zeros to be ui32.	ui32
Y : Length of Data	Number of Click Bytes at DRAM location Z. Also leading Zero stuffed.	ui32
Z : Memory pointer to data	*DRAM pointer	ui32
T :Operating system will append the system time here, Apps don't have to.	Application does not use this field.	ui64
}		

Figure 13 : Clickstream Event Type Data Structure

The Clickstream Kernel will receive the OS Event sent by the application. It will use the information provided in the X, Y, and Z parameters to grab the Clickstream data out of it's temporary location in DRAM and into the Clickstream double buffer area. The Clickstream Kernel will concatenate this Clickstream data at the end of the buffer it is currently maintaining. It performs a level of memory management on the double buffers to maintain current pointers of free memory.

3.5.3. Clickstream API Source Code Example

The following is a source code example of a miscellaneous application making a Clickstream journalling call:

```
// *******General Header Code**********************************************************
//Apps would place this first part of code once in a header area,
//or make sure it was in a #include

    #define CLICK                          // make it easy to pull the code out at compile time
    #ifdef CLICK
    {
    #define kEt_clickStream   0x00008000 //all apps who wish to journal use this #define, it will
                                          // live in the PTVTYPES.h file by end of year.
    #define kEd_clickEntry    0          // event data for normal click stream event
    #define CLICK_BUFFER_SIZE 0x20       //local storage of one or two clicks, this size will be
                                          //slightly application dependent, but 16 to 32 bytes
                                          //should be sufficient for most applications.
    #define APP_ID 0x0000                // Application Identifier should be defined here.
                                          // IDs for all applications are listed in Figure 3

    void *dptr;                           //declare a pointer to local Click data storage
    }
    #endif

// **    End of General Header Code**********************************************************


// ******* Code for each key click occurances                    ****************************************

#ifdef CLICK                              // make it easy to pull the code out at compile time
{
dptr = pk_Galloc( CLICK_BUFFER_SIZE ); //Application needs to grab some amount of
                                       //DRAM for temporary storage of Clicks locally.
sprintf((ui8 *)dptr,"clickstream stuff");    //place Click data into memory at the *dptr location.
//This is an example of placing a string of ASCII text at this location. App developers will be
responsible for writing code to place a data struct as outlined in Section 3 into memory at *dptr.

pk_PostEvent(kEt_clickStream | kEd_clickEntry, APP_ID,(ui32)strlen(dptr),(ui32)dptr);
}
#endif

// ******* End of code for each key click occurances             ****************************************
```

Developer Note:

The code associated with each Key Click will be responsible for Malloc'ing memory for each Click, and the Clickstream Kernel will then De-allocate the

memory assigned with every Click. Because of this it is imperative that the sending application check to make sure that the clickstream application is up and running in background mode. The sending application can do this by querying the Application Manager for an application by the name of "CLICKSTREAM". This call will also return to the calling application the system event que of the Clickstream Processor. If the Clickstream Processor is NOT running in the background to De-allocate this memory, there will be memory leaks associated with each attempted journalling call by the sending application.

3.5.4. Clickstream Upload

Once a Clickstream buffer has been filled, or the Clickstream Kernel has decided an upload of data is appropriate for other reasons (timed upload, low system utilization, commanded upload, etc.) the buffer will be formatted, compressed, and then uploaded through the system to the Sybase Staging Server.
The Clickstream upload takes place over the system messaging or Level 1 pass-through protocol. The data protocol is uni-directional UDP/IP and so there will be application level acknowledgements from the Staging Server during the upload process. The STB application will use the PowerTV system messaging APIs to send data packets upstream.

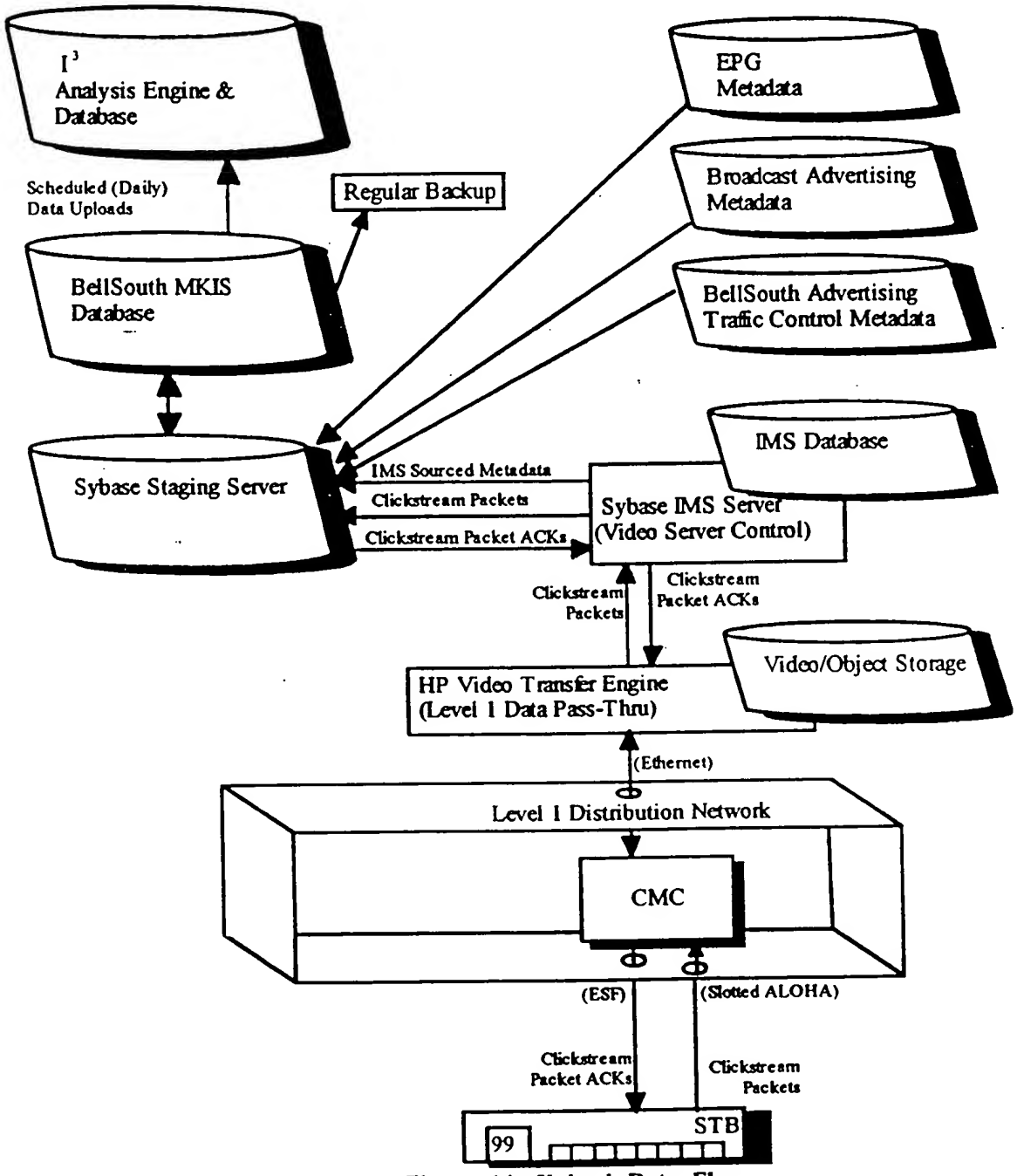


Figure 14: Upload Data Flow

3.5.4.1. System Upload Control / Load Distribution

There will be several ways for the upload of Clickstream information to be controlled. A broadcast transaction set will be used to set up the upload control method used, or uploads can be commanded by the system. In implementation, this means MKIS or Staging Server will source a set of transactions delivered over the Level 1 pass-through messaging to perform this upload control. The data associated with each STB will be stored in Non-Volatile Memory. Because of the small footprint of the data, and the adverse effects that would be caused if the upload control data parameters were not present within the STB, this data will reside in NVM.

3.5.4.1.1. STB Group Assignment for Clickstream Uploads

In order to assign STBs to a uniformly distributed set of groups during the trial period, the 48 bit STB MAC address can be used to randomly distribute boxes within groups in the system. The Clickstream Kernel will use the least significant 5 bits of the MAC address to determine the Clickstream group in which it belongs. This will define 32 groups within the system. This way a separate addressable transaction data packet for group assignment does not have to be defined, transmitted, and received during the trial period. This is a short term solution, and would have to be modified to an addressable group assignment methodology concurrent with large scale deployment.

STB MAC Address						
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	By 7 6 5	[illegible]
						Group Assignment

Figure 15: Clickstream Group Assignment

3.5.4.1.2. Upload Control Broadcast Message

This one message depending on how it is configured with data will allow for the following control:

1) Cyclic Upload Control to Groups
2) Cyclic Upload Control to all STBs
3) Upload on Command/Polling Control (addressable only)
4) Suppression of Upload on buffer full
5) Collection Control On/Off addressable

TCP/IP Protocol Header	**Adaptation Type**	**Protocol Discriminator**
40 Octet Destination: CMC Sender: VTE	1 Octet 0x02 = TCP	1 Octet (Fixed for BLS Trial) 0x80
VTE inserted -------->		

Adaptation Data	**Message_Id**	**Request_Id**	**Address_Mode**
2 Octet Length (Octets to follow, Not including Trailer)	1 Octet (Fixed) 0x60	2 Octet Not Used	1 Octet 0x01 = Addressable 0x04 = Broadcast
VTE inserted -------->			**Level 2-Defined ----->**

Destination_Address	L2_DataCount	Data_Type	VSP_Id
6 Octet • MAC Address or • 0xFFFFF..... = Broadcast	2 Octet # Octets to Payload End Variable	1 Octet 0x00 = User Data	1 Octet 0x01 = BIMS
Level 2-Defined -------->			

Clickstream Upload Control (Transaction_Code and Transaction_Payload)			
Octet#	Contents		
T 0	Transaction Code MSB = 0x80		
T 1	Transaction Code LSB = 0x10		
0	Clickstream Processor Version Number		
1	Global Flag (b1)	Addressable Flag (b1)	Group Address - Denotes the group of STBs to field this transaction (b5)
2	Collection On/Off Key - Will turn Clickstream collection On/Off to a STB or Group of STBs (non-Global only)		
3	Perform Upload Now Key - Will perform an upload on command. Will only upload on command if Global Flag is NOT set.		
4	Suppress Upload on Buffer Full - Will keep the STB or Group from uploading when buffer is full. The STB or Group will only upload on it's appointed upload cycle.		
5	Upload_Cycle_Definition - A STB will have 1 to 4 possible upload cycles defined. This will define any one of those cycles.		
6	Cycle First Occurrence Start Time (Total b48) - Year (b8) Defines the time of the first upload in cycle.		
7	Cycle First Occurrence Start Time - Month (b8)		
8	Cycle First Occurrence Start Time - Day (b8)		
9	Cycle First Occurrence Start Time - Hour (b8)		
A	Cycle First Occurrence Start Time - Minute (b8)		
B	Cycle First Occurrence Start Time - Second (b8)		
C	Upload Duration (Total b24) - Hours(0-24) (b8) Defines a duration of time over which the STB randomizes upload start time.		
D	Upload Duration - Minutes(0-59) (b8)		
E	Upload Duration - Seconds(0-59) (b8)		
F	Cycle Time (Total b32) - Days (0-14) (b8) Defines the periodicity between uploads. This is the mean time between uploads.		
10	Cycle Time - Hours(0-24) (b8)		
11	Cycle Time - Minutes(0-59) (b8)		
12	Cycle Time - Seconds(0-59) (b8)		
0x13 to 0x27	reserved		

TCP/IP Protocol Trailer
CRC Checksum, etc
VTE inserted -------->

Figure 16: Upload Control Transaction Definition

Data Dictionary:

The Clickstream upload cycle consists of several parameters which define a start time and a cycle on which the uploading of data happen. The "first occurrence" defines a starting time in history from which the cycle runs. The "cycle time" defines the amount of time which elapses between periods of the upload cycle. When a cycle is complete the "upload duration" time starts, and the STB Clickstream Processor will randomize an exact upload time within the Upload Duration. This will distribute the network load relatively evenly over the entire Upload Duration period.

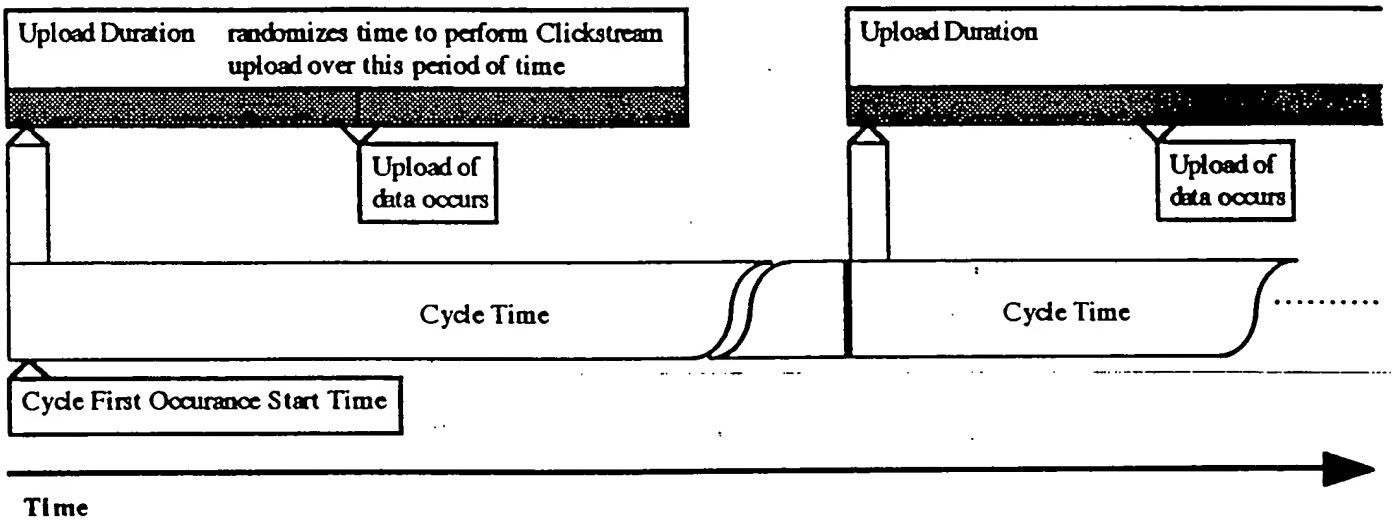


Figure 17: Upload Cycle Diagram

An example of the use of these parameters would be to define a period of time every day for STBs to upload data. BellSouth has a requirement to perform analysis of uploaded data on a daily basis. This should be available every morning towards the beginning of the work day. Peak usage of broadcast Prime Time and of Interactive services will typicly be from 7pm until Midnight. This would be a time period when uploads should be inhibited to limit the burden on the Level 1 distribution network, which will be busy with exclusive session setups and tear-downs. Beginning at Midnight, Clickstream uploads would begin. In order to have all STBs upload before 8am, and given the number of upload groups within the system to be 32, each group would perform uploads over a 15 minute period. To achieve this upload cycle:

Cycle_First_Occurance_Start_Time	= 12:00am Jan 1, 1995 + "X" * 15 minutes. Each upload group would have this parameter staggered by 15 minutes.
Cycle Time	= 24 hours
Upload Duration	= 15 minutes

A total of 4 upload cycles will be able to be defined for each group of STBs. This would allow for weekly uploads, or any other combination of cycles to work around peak L1 load times.

3.5.4.2. Communications Procedures for Upload

The STB will initiate the upload process. When the STB has determined it is appropriate to perform an upload of Clicsktream data (for any of the reasons covered in section 3.5.4), it will lock the DRAM buffer actively being filled, it will compress the buffer using the LZW compression utility, and it will perform the upload of the Clickstream buffer over the L1 Pass-Through network.

The STB initiates reverse path L1 Pass-Through by calling the "sess_MessageRequest" PowerTV API. Within the parameters of this API are the addressing Mode, the destination address, the size of the "message" packet to send, and a pointer to the data in memory to send upstream.

rc sess_MessageRequest (ui8 mode, ui8 address[6], ui16 sp_count, ui8 *sp_data);

From this point, the PowerTV operating system and lower level drivers will handle the uploading of the message, including the insertion of the UDP/IP protocol layer and CRC-32 transport trailer.

There WILL be a limit to the size of the data allowed in a single reverse path L1 Pass-Through transaction. Today, this is not explicit in PowerTV documentation, but in disclosure documentation it is stated as **252 Octets**. Because most Clickstream uploads will require more than this, the Clickstream Processor will be making sequential calls to this MessageRequest API to complete a single upload process.

There are several different addressing modes available in L1 Pass-Through. The one that Clickstream Processor will be using will identify a VSP as the destination of these data packets. That address_mode is 0x03. The address following this address_mode is a 2 Octet SP_ID (short for VSP identifier). It should be right justified and 0 filled into the 6 byte field available. The sp_count and *sp_data are simply the size and pointer to the data to be uploaded. All of the data to be uploaded appears as "Payload" to the STB, the signalling network, the CMC, and the Event Capture mechanism on the Staging Server back end Level 2 system.

IP Header	UDP Header	Adaptation Type	Protocol Discriminator
20 Octet Destination: CMC As defined in RFC 791	8 Octet As defined in RFC 768	1 Octet 0x01 = UDP	1 Octet (Fixed for BLS Trial) 0x80
PowerTV OS inserted -------->			

Adaptation Data	Message_Id	Request_Id	Address_Mode
2 Octet 0x0000 or sequential counter	1 Octet (Fixed) 0x60	2 Octet Not Used	1 Octet Address of VSP 0x03 = "SP_ID"
PowerTV OS inserted -------->			Application-Defined>

Destination_Address	L2_DataCount	IP Address of AS	Service ID Length
6 Octet • Address of VSP	2 Octet # Octets to Payload End Variable	4 Octet (Fixed) • IP Address of App Server	4 Octet TBD
Application-Defined------>			

Service ID Value	STB MAC Address	Data_Type	VSP_Id
4 Octet TBD	6 Octet • Hardware Address of STB	1 Octet 0x00 = User Data	1 Octet 0x01 = BIMS
Application-Defined------>			

Clickstream Upload Data Packet (Transaction_Code and Transaction_Payload)	
Octet#	Contents
T 0	Transaction Code MSB = **0x80**
T 1	Transaction Code LSB = **0x18**
0	Clickstream Processor Version Number
1	Upload Sequence Number
0x02 thru 0xFA	**Clickstream Upload Buffer Data Structure** (as outlined in Figure 3) The data structure is broken up into as many reverse path transactions as are neccessary to perform the complete upload of data.

UDP Transport Trailer		
Control	Length	CRC-32
2 Octet (Fixed) 0xFFFF	2 Octet : # Octets from IP Header to Payload End Variable Length	4 Octet CRC-32 Calculation
PowerTV OS inserted -------->		

Figure 18 :Upload Transaction Definition

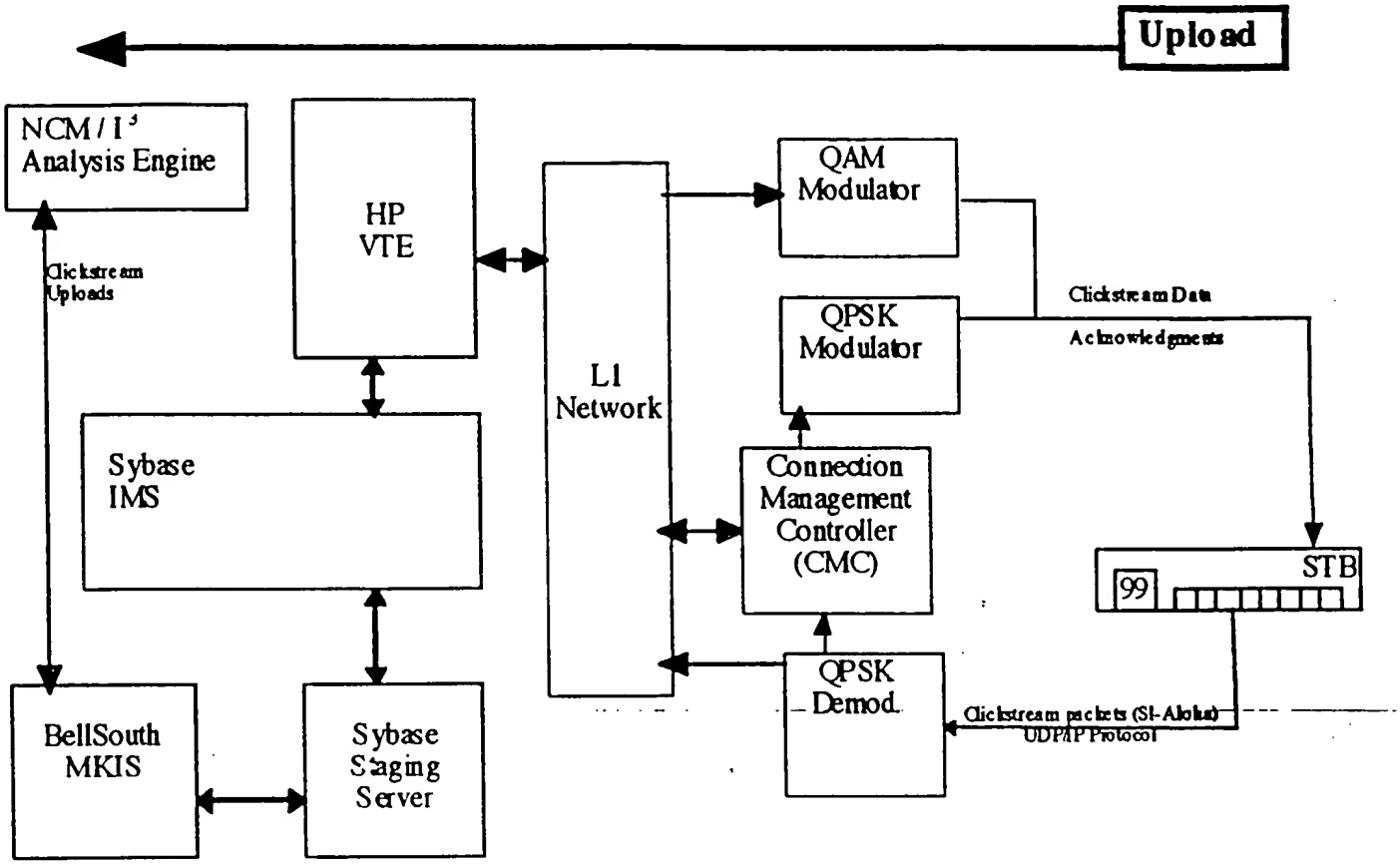


Figure 19: Upload Data Flow 2

3.5.4.3. System Acknowledgements on Clickstream Buffer Uploads

The Event Capture Procedure running on Staging Server will be responsible for formulating and sending some level of data acknowledgements to each STB engaged in the upload process. The details of the acknowledgement process is still a work-in-progress. The acknowledgements will go out as addressable downstream Level 1 Pass-Through transactions. They will enter the STB through the UDP/IP protocol stack, and be handed to the Clickstream Processor as a PowerTV Event which the Processor has registered an interest in.

TCP/IP Protocol Header	Adaptation Type	Protocol Discriminator
40 Octet Destination: CMC Sender: VTE	1 Octet 0x02 = TCP	1 Octet (Fixed for BLS Trial) 0x80
VTE inserted -------->		

Adaptation Data	Message_Id	Request_Id	Address_Mode
2 Octet Length (Octets to follow, Not including Trailer)	1 Octet (Fixed) 0x60	2 Octet Not Used	1 Octet 0x01 = Addressable
VTE inserted -------->			Level 2-Defined ----->

Destination_Address	L2_DataCount	Data_Type	VSP_Id
6 Octet • MAC Address	2 Octet # Octets to Payload End Variable	1 Octet 0x00 = User Data	1 Octet 0x01 = BIMS
Level 2-Defined -------->			

Transaction_Code	Transaction_Payload
2 Octet Defines TX_Payload 0x801F	Data Acknowledgement Field (TBD) Application Level Data
Level 2-Defined -------->	

TCP/IP Protocol Trailer
CRC Checksum, etc
VTE inserted -------->

Figure 20 : Acknowledgement Transaction Defined (As Sent by Staging Server)

3.5.4.3.1.UDP/IP Protocol Header

The UDP/IP protocol stack within the STB will be handling insertion of the UDP/IP headers and transport trailer for L1 Pass-Through transactions transmitted upstream. The header is directly from the RFC 791 and RFC 768 specifications. They are as follows:

IP Header				
IP Version	Header Length	Type of Service	Total Length	
Identification			Flags	Fragment Offset
Time to Live	Protocol		Header Checksum	
Source IP Address				
Destination IP Address				

UDP Header	
Source Port	Destination Port
Length	Checksum

Figure 21 : IP and UDP Header Definitions

3.5.4.3.2.Data Compression on Upload

The data compression algorithm LZW has been selected by PowerTV as a utility they will be providing to applications from the Operating System at some point in the future. They will be using LZW primarily for decompression of large BLOBs , graphics files, data objects, etc. The Clickstream processor will also use this algorithm to compress the Clickstream buffer before data is uploaded through the system.

The LZW Compression algorithm will be treated in detail in the next release of this document.

3.5.4.3.3.Level 1 Pass-Through Mechanism to IMS

This is currently a work in progress. APIs have been defined but have not been delivered to BellSouth.

4. IMS Server Functions

4.1. IMS Sourced Logs

The IMS will be the source for content Metadata regarding all session based services, and in addition will act as a subset of the Clickstream information gathered on the session based services offered.

4.2. Pass Through on Clickstream

This interface does little more than act as a gateway for the reverse path L1 Pass-Through Transactions to flow. It does not manipulate the contents of the L1 Pass-Through Transaction. The IMS will wait on an open connection to the "Event Capture" process running on the Staging server, and will pass Clickstream data over that connection as it is received. The interface to L1 Pass-Through on the Staging server side is called by the following function call.

```
Int     IMS_P_L1_Pass_Through ( Int   Size, VarBinary   Payload);
```

5. Metadata Formats

Content Metadata is information in an encoded form which describes the programming shown on the network and the time and broadcast or cable network on which the programming was shown. In the term "Content" we are including programming of virtually any type which is transmitted to the subscriber's Set-Top Box (STB). There are a number of sources for this information and it will be of utmost importance to manage this information appropriately. For this system, we will be interested in the merging of Content Metadata into our Clickstream data primarily for the analysis of Broadcast services provided by the Cable Television Application (Cable App) and VOD/NVOD usage. We will also be interested in the numbers of STBs tuned to other interactive services when analyzing ratings and general viewing habits of our subscriber population. The merging of Content Metadata into our Clickstream data adds the relevant information upon which most of our analyses will be based.

The sources of Content Metadata for this system are shaping up to be:

1) **EPG System:** General broadcast programming content.

2) **Broadcast Advertising:** Advertising content occurrences on most national , local, and cable broadcast networks. This will not include some spot advertisements generated by local broadcasters, and will not include any spot advertisements generated within the BellSouth system.

3) **IMS System Logs :** The IMS system will Log all stream occurrences, and the STB or STBs associated with the stream. This will primarily be in the form of VOD, NVOD, and interactive application execution (i.e. the Navigator).

4) **Advertising Traffic Controller:** This system is still in very formative stages at the time of this writing. It will be a BellSouth system enabling the control of advertisement traffic through the BellSouth system throughout the interactive video trial. This system will hand off advertisement Metadata which is generated in-house by the insertion of advertisements in both broadcast and session-based services.

There will also be system globally defined Metadata. There are lookup tables by which all sorts of other contextual information may be determined. These tables should be global to the BellSouth System were possible. Many, if not all of these tables are defined in this section.

The Clickstream system will attempt to resolve data formats coming from the various sources by defining a global data storage format in the MKIS system, and resolving any variations from that global format with the Metadata providers before the system comes on-line. If all variations cannot be resolved at their sources, an additional parsing mechanism will have to be implemented.

In addition, there will be Content which is reported by several different sources. We will have to resolve these "Duplicate Occurrences" for our Content data to be as accurate as possible.

5.1. Globally Defined Metadata Formats

Content Metadata is logged in a single main tabular form. The Metadata IDs are then resolved by several supporting data tables. In the MKIS system this data structure will take the form of a relational database model.

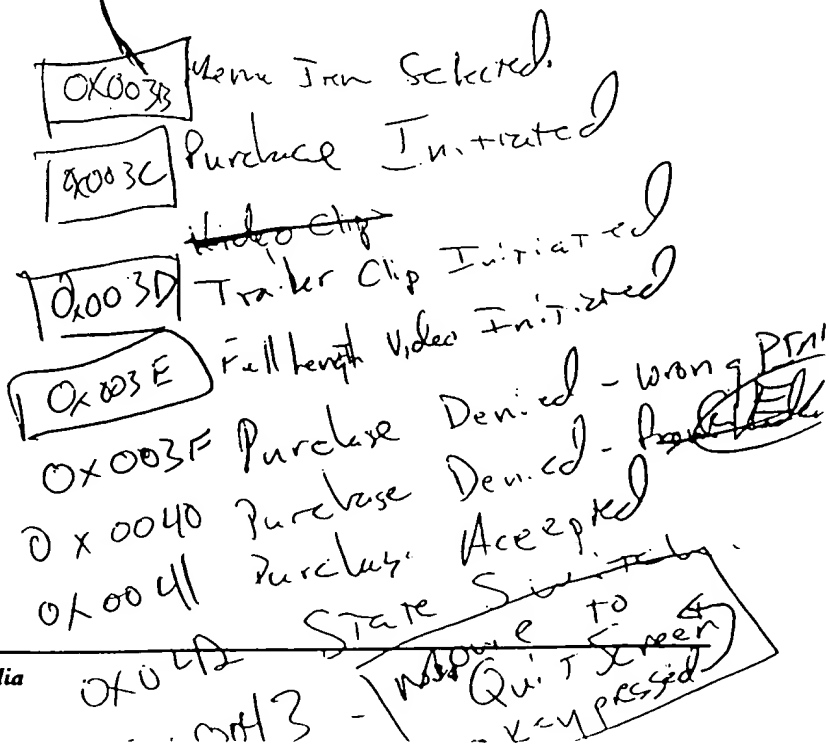
5.1.1. Global Event ID Definitions

A system global method for defining Event IDs is needed by every application to provide a context by which the event was important enough to be journaled. This could be as simple as a broadcast channel change by pressing the Chan Up key. It could be a passive change in the content shown on the network. All of these event types are gathered into a single table for purposes of easing the data analysis process. No single application will make use of

every possible event type. In most cases it will benefit the system overall for applications to make use of as little event types as possible, while still extracting the usage information necessary for analysis.

EVENT DEFINITIONS	
Content Related Events	
0x0000	Passive Change in Content
Direct Key Presses	
0x0001	TV <> iTV Pressed
0x0002	Power Pressed
0x0003	One (1) Pressed
0x0004	Two (2) Pressed
0x0005	Three (3) Pressed
0x0006	Four (4) Pressed
0x0007	Five (5) Pressed
0x0008	Six (6) Pressed
0x0009	Seven (7) Pressed
0x000A	Eight (8) Pressed
0x000B	Nine (9) Pressed
0x000C	Zero (0) Pressed
0x000D	Channel Up Pressed
0x000E	Channel Down Pressed
0x000F	Volume Up Pressed
0x0010	Volume Down Pressed
0x0011	Last Channel Pressed
0x0012	Favorite Channel Pressed
0x0013	Guide Key Pressed
0x0014	Theme Key Pressed
0x0015	Display Key Pressed
0x0016	Mute Key Pressed
0x0017	A Key Pressed
0x0018	B Key Pressed
0x0019	C Key Pressed
0x001A	Info Key Pressed
0x001B	Backup Key Pressed
0x001C	Main Menu Key Pressed
0x001D	Up Arrow Key Pressed
0x001E	Down Arrow Key Pressed
0x001F	Right Arrow Key Pressed
0x0020	Left Arrow Key Pressed
0x0021	Enter Key Pressed
0x0022	Play Pressed
0x0023	Rewind Pressed
0x0024	Pause Pressed
0x0025	Record Pressed
0x0026	Fast Forward Pressed
0x0027	Stop Pressed
Application/State Switching Related	
0x0028	AC Power ON
0x0029	Application Switch (Normal)
0x002A	Application Switch (Abnormal)
0x002B	Application Terminated (Normal)
0x002C	Application Terminated (Abnormal)
0x002D	Soft Power OFF
0x002E	Soft Power ON
0x002F	OFF State Polling Event
General	
0x0030	Direct Channel Change
0x0031	Mute
0x0032	Un-Mute
0x0033	Volume Change Below 50%
0x0034	Volume Change Below 25%
0x0035	Volume Change Below 10%
0x0036	Volume Change Above 50%
0x0037	Volume Change Above 25%
0x0038	Volume Change Above 10%
0x0039	Change to Interactive Mode
0x003A	Change to Broadcast Mode

Figure 22: Global Event ID Definitions



5.1.2. Timestamp Format

The timestamp format for all Clickstream events throughout the system will be based on **a 6 byte (48 bit) subset of the ANSI C and C++ standard TIME_T data structure.** It will identify a unique time since the year 1900 down to a one second resolution. Some of our content Metadata providers will only be able to give us data down to a one minute resolution, if this is the case, these Metadata records will have to be appended or modified defaulting the "second" field to 00. The following is an overview of the TIME_T data structure.

	<time.h>		
int	tm_sec	seconds after the minute	(0,61)
int	tm_min	minutes after the hour	(0,59)
int	tm_hour	hours since midnight	(0,23)
int	tm_mday	day of the month	(1,31)
int	tm_mon	months *since* January	(0,11)
int	tm_year	years since 1900	(0, 256)

Figure 23: Global Time Format

not used int tm_wday days since Sunday (0,6)
not used int tm_yday days since January (0,356)
not used int tm_isdst Daylight Savings Time Flag

The entire data structure as defined in the ANSI C spec is a full 9 Bytes. The last 3 bytes deal with defining the day of the week, the day of the year, and a daylight savings time flag. These could very easily be extrapolated after event generation in the MKIS system if needed, and it is not clear that they would be needed for any of the known analyses. The elimination of these three bytes from the Clickstream event decreases storage and bandwidth needs.

5.1.3. Global Broadcast Channel Identifiers

The following table defines a coding scheme for national and local broadcasters which will be carried on the BellSouth network during the Residential Broadband Trial. This list may grow as channels are added, but the identifiers will remain unique and constant for each channel. Any application journaling off events which occur while subscribers are viewing broadcast television will be able to identify the network carrying the programming content by using a subset of this table. In this way channel lineups can be changed over the life of the residential broadband trial, (and presumably a larger scale BellSouth services deployment), and the identifier for a network would stay the same. Data repositories will be relieved of the task of mapping an ever-changing channel number to a network.

Broadcast Channel Identification Table	
0x0100 to	
0x011F	**News/Talk Shows**
0x0100	CNN
0x0101	Headline News
0x0102	The Weather Channel
0x0103	CNBC
0x0104	CSPAN
0x0105	CSPAN-2
0x0106	America's Talking
0x0107	Talk Channel
0x0108	Court TV
0x0109	The Crime Channel
0x010A	National Empowerment TV
0x0120 to	
0x013F	**Sports**
0x0120	ESPN
0x0121	ESPN-2
0x0122	SportSouth
0x0123	The Golf Channel
0x0124	Classic Sports Network
0x0125	Prime Network
0x0126	NewSport
0x0140 to	
0x015F	**Music**
0x0140	MTV
0x0141	VH-1
0x0142	Country Music Television
0x0143	The Nashville Network
0x0144	The Box
0x0145	Video Jukebox
0x0146	MOR Music TV
0x0147	Music Choice
0x0160 to	
0x017F	**Shopping**
0x0160	QVC
0x0161	QVC-2
0x0162	Home Shopping Network
0x0163	TV Macy's
0x0164	Catalog 1
0x0165	S, The Shopping Network
0x0166	Cupid Network
0x0180 to	
0x019F	**Movies**
0x0180	Turner Classic Movies
0x0181	American Movie Classics
0x0182	TNT
0x0183	Popcorn Channel
0x01A0 to	
0x01BF	**Religious**
0x01A0	Faith & Values Channel
0x01A1	The Inspirational Network
0x01A2	Trinity Broadcasting Network
0x01A3	Eternal World TV Network
0x01A4	The Gospel Network
0x01C0 to	
0x01DF	**Health & Self-Improvement**
0x01C0	Lifetime
0x01C1	Cable Health Club
0x01C2	The Health Channel
0x01C3	Parent Television
0x01C4	Recovery Network / The Wellness Channel
0x01E0 to	
0x01FF	**Cultural/Ethnic**
0x01E0	A &E
0x01E1	BRAVO
0x01E2	El
0x01E3	Ovation
0x01E4	BET
0x01E5	BET International
0x01E6	BET on Jazz
0x01E7	World African Network
0x01E8	Univision
0x01E9	Galavision
0x01EA	Telemundo
0x01EB	GEMS
0x01EC	International Channel
0x0200 to	
0x021F	**Educational**
0x0200	The Discovery Channel
0x0201	The History Channel
0x0202	The Learning Channel
0x0203	Mind Extension University
0x0220 to	
0x023F	**Kids**
0x0220	Nickelodean
0x0221	Cartoon Network
0x0240 to	

0x025F	General
0x0240	TBS
0x0241	The Family Channel
0x0242	USA Network
0x0243	fX
0x0244	WGN
0x0245	WWOR (NY)
0x0246	WPIX (NY)
0x0247	KTLA (LA)
0x0248	KTVT
0x0260 to	
0x027F	**Specialty**
0x0260	Sci-Fi Channel
0x0261	Comedy Central
0x0262	Sega Channel
0x0263	Nostalgia Television
0x0264	Americana Television Network
0x0265	The Collector's Channel
0x0266	TV Food Network
0x0267	The Travel Channel
0x0268	Jones Computer Network
0x0269	The Game Show Channel
0x026A	The Ecology Channel
0x026B	Home & Garden TV
0x026C	Kaleidoscope: America's Disability Channel
0x026D	The Military Channel
0x026E	The Navy Channel
0x026F	The Singles Channel
0x0270	The Musician Channel
0x0271	Romance Classics
0x0280 to	
0x02AF	**Premium Channels**
0x0280	HBO 1
0x0281	HBO 2
0x0282	HBO 3
0x0283	Cinemax 1
0x0284	Cinemax 2
0x0285	The Movie Channel
0x0286	Showtime 1
0x0287	Showtime 2
0x0288	Showtime Espanol
0x0289	Showtime Family
0x028A	Showtime Film
0x028B	Showtime Comedy
0x028C	Showtime Action
0x028D	The Disney Channel
0x028E	Adam & Eve
0x028F	Spice
0x0290	Spice 2
0x0291	TheatreVision
0x0292	Playboy
0x0293	Request TV
0x0294	Viewer's Choice
0x0296	Your Choice TV
0x0297	Cable Video Store
0x0300 to	
0x033F	**Local Broadcast Providers**
0x0300	WSB
0x0301	WAGA
0x0302	WGTV
0x0303	WXIA
0x0304	WTBS
0x0305	WPBA
0x0306	WATL
0x0307	WGNX
0x0308	WVEU
0xFFE0 to	
0xFFFF	**Corner Cases / Error Conditions**
0xFFFE0	Provider To Be Defined
0xFFFE1	Provider Unknown
0xFFFFF	General Error

5.2. *Content Metadata Formats*

Metadata regarding programming content provided in the BellSouth system is necessary to tie events which occur on the STB to what content the event may have been linked with. "How many subscribers changed the channel during every Coca-Cola commercial?", "Do subscribers channel surf during some programs more than others?", etc. All possible queries against our usage data will be enabled only by our ability to tie content to the services provided. The following is a general format for all content Metadata. A main table (table 1) ties a piece of programming content to a particular channel identifier between an accurate start and end time. The table 2 simply supports the ID scheme used in table 1 to minimize data storage and transmission capacities. Other Channel and Time encoding schemes are defined in section 5.1 above.

	Content ID	Channel ID	Start Time	End Time
Examples	0x012345FF	0x002F	0x0000FEDCBA00	0x0000FEDCBF98
	0x01234567	0x002F	0x0000FEDCBA98	0x0000FEDCBFFF

Table 1

	Content ID	Content Unique Descriptor	Content Type
Examples	0x01234568	"MURPHY BROWN, #39"	0x04
	0x01234567	"HERBIE RIDES AGAIN"	0x0F

Table 2

	Content Type	Content Type Descriptor
Examples	0x04	"Situation Comedy"
	0x05	"Movie, Drama"
	0x06	"Advertisement"
	0x0F	"Movie, Comedy"

Table 3

Global Content Data Formats:

Content ID : 32 Bit unique identifier for content.
Channel ID : 16 Bit unique identifier for programming network.
Start Time: 48 Bit standard C / C++ "TIME_T" data format for description of time.
End Time: 48 Bit standard C / C++ "TIME_T" data format for description of time.
Content Unique Descriptor: Variable length character string describing programming content.

5.2.1. Electronic Program Guide Metadata Format

This is an example of the types of data we will expect to be sourced by Prevue Guide during the Residential Broadband Trial period. Data formats can be expected to be more complex than that represented here. These data formats are expected to finalize over the next month.

Prevue Guide Metadata is replicated forward from the Prevue Guide server to the Staging Server through a Client / Server replication RPC.

	Content ID	Channel ID	Start Time	End Time
Examples	0x012345FF	0x002F	0x0000FEDCBA00	0x0000FEDCBF98
	0x01234567	0x002F	0x0000FEDCBA98	0x0000FEDCBFFF

Table 1

	Content ID	Content Unique Descriptor	Content Type
Examples	0x01234568	"MURPHY BROWN, #39"	0x04
	0x01234567	"HERBIE RIDES AGAIN"	0x0F

Table 2

Electronic Program Guide Content Data Formats:

Content ID : 32 Bit unique identifier for content.
Channel ID : 16 Bit unique identifier for programming network.
Start Time: 48 Bit standard C / C++ "TIME_T" data format for description of time.
End Time: 48 Bit standard C / C++ "TIME_T" data format for description of time.

5.2.2. Advertisement Metadata Format:

The format of advertisement data to feed into the MKIS would take the form of two tables. The first table would be the time/channel grid relating an advertisement identifier to a channel ID at a particular start and end time. The second would be a table relating advertisement identifiers to the actual advertisement content as described in a character string, the advertisement identifiers would describe an advertisement content for the life of the system,(i.e. IDs would not be recycled). The reporting agency would use our existing Channel ID metadata, and any other BellSouth standard encoding scheme for their content where possible.

	Advertisement ID	Channel ID	Start Time	End Time
Example	0x01234567	0x002F	0x0000FEDCBA98	0x0000FEDCBAFF

Table 1

	Advertisement ID	Advertisement Unique Descriptor
Example	0x01234567	"Coca Cola Corporation Ad #1357"

Table 2

Advertising Data Formats:

Advertisement ID : 32 Bit unique identifier for advertising content.
Channel ID : 16 Bit unique identifier for programming network.
Start Time: 32 Bit standard C and C++ "TIME_T" data format for description of time.
End Time: 32 Bit standard C and C++ "TIME_T" data format for description of time.
Advertisement Unique Descriptor: Variable length character string describing advertisement.

5.2.3. IMS Log Metadata Formats

5.2.4. BellSouth Advertising Traffic Control Metadata Formats

The format of in-house advertisement data to feed into the MKIS would be identical to third-party sourced data.

	Advertisement ID	Channel ID	Start Time	End Time
Example	0x01234567	0x002F	0x0000FEDCBA98	0x0000FEDCBAFF

Table 1

	Advertisement ID	Advertisement Unique Descriptor
Example	0x01234567	"Coca Cola Corporation Ad #1357"

Table 2

Advertising Data Formats:

Advertisement ID : 32 Bit unique identifier for advertising content.
Channel ID : 16 Bit unique identifier for programming network.
Start Time: 32 Bit standard C and C++ "TIME_T" data format for description of time.
End Time: 32 Bit standard C and C++ "TIME_T" data format for description of time.
Advertisement Unique Descriptor: Variable length character string describing advertisement

6. Staging Server Functions

The Staging Server is a Sybase Conceptual Database entity which will serve many purposes in the trial. The main purpose in it's conception is a method to distribute many storage and communications processes off-line from the IMS which will be very busy with processor-intensive and time-critical actions controlling the VTE and handling signaling sessions. The actual hardware on which this system will reside may depend on performance we experience when the system comes on-line. In-depth documentation on the many functions of the Staging Server will be handled by Sybase eventually when these designs come to a state of completion. This section will act as a rough overview of the Staging Server processes involved in the Clickstream System.

6.1. Upload Communications Procedure: Event Capture

6.1.1. RPC Endpoint

The Staging Server will act as the endpoint of the initial upload procedure of STB Clickstream information. When IMS receives a Level 1 Pass-Through packet with Clickstream data, the packet will be addressed to a particular process running on the Staging Server. This "Event Capture" prococess rung on the Staging Server will maintain an open connection with the Level 1 Pass-Through running on IMS.

6.2. Persistent Storage

The Event Capture process will provide for flat-file storage of da as is is delivered to the Staging Server over upstream communications. Data will then be decompressed and parsed from this flat file.

6.3. Content Merge with Metadata

Clickstream data must be brought together with contextual information on programming and advertising from both broadcast and interactive services. If this is done so in an effective manner, then analyses on the resultant data will provide an enormous amount of value to BellSouth. The processes of merging the data could be done in a number of different ways. The method advocated in this design document reflects a consideration for availability and format of data, system bandwidths, processing capabilities, storage requirements, and analysis capabilities.

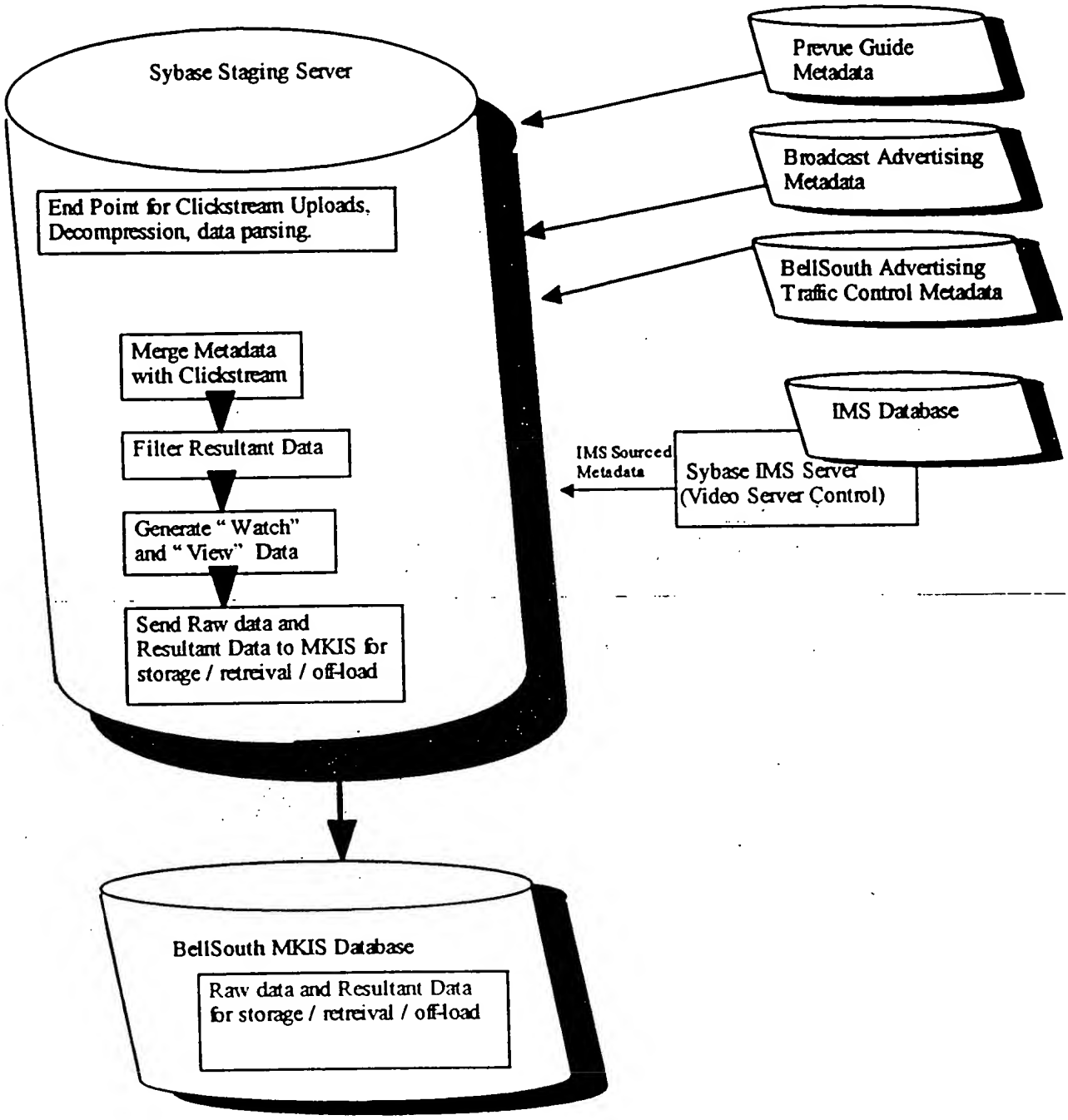


Figure 24 : Staging Server Overview

The merging methods break down into three parts.

1) First a complete timeline is constructed for each Set-Top Box for an entire 24 hour "Day" period. This is done by inserting content identifiers into the Clickstream data, and by generating additional Clickstream data when content has changed and the subscriber continued to view the same programming channel or service.

2) Filtering is done to "weed out" any extraneous data. This part of the merging process will have to evolve over time as we learn more about the types of data which have proved to be valuable, and those which haven't.

3) The timeline is further analyzed to provide a second timeline. This second data output is a list of content items "watched" by the STB and a list of content items "viewed" by the STB. These lists are generated based on decision criteria provided by BellSouth Marketing and Advertisement departments.

These steps are repeated for each Set-Top Box in the system.

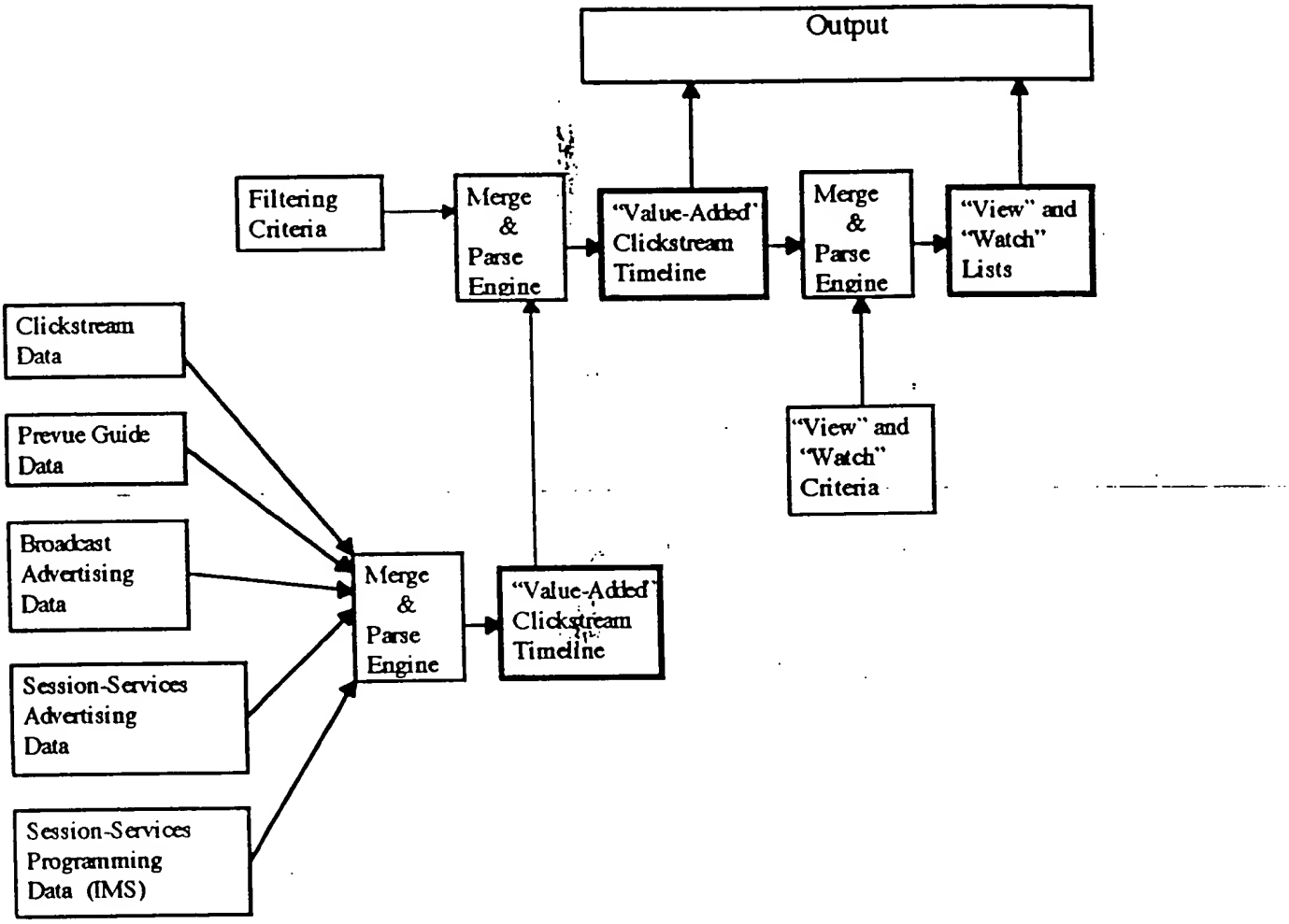


Figure 25: Merging & Parsing Overview

Merging and parsing should work on 24 hour segments of data at once to minimize the occurrence of un-resolved events. This is to say that Clickstream Events are in and of themselves just pieces, and to do analyses on just several hours of data will leave the determination of programs being "watched" and "viewed" at the endpoints unresolved.

Source: STB

Clickstream Data

		Timestamp	Application ID	# Bytes to Follow	Application Specific Data Event ID	Channel ID
1	Event Record	3:59:30pm	Cable App.	4 bytes	Soft Power On	ABC
2	Event Record	4:00:00pm	Cable App.	4 bytes	Channel Up	NBC
3	Event Record	4:03:17pm	Cable App.	4 bytes	Channel Up	TBS
4	Event Record	4:06:25pm	Cable App.	4 bytes	Channel Dwn	NBC
5	Event Record	4:15:45pm	Cable App.	4 bytes	Channel Up	TBS
6	Event Record	4:55:45pm	Cable App.	4 bytes	Soft Power Off	None

EPG

"Prevue" Guide Data

		Content ID	Channel ID	Start Time	End Time
1	Content Record	National News	ABC	3:30:00pm	4:00:00pm
2	Content Record	Murphy Brown	NBC	3:59:00pm	4:30:00pm
3	Content Record	Simpsons	NBC	4:30:00pm	4:59:00pm
4	Content Record	N.G.Explorer	TBS	3:05:00pm	4:05:00pm
5	Content Record	Andy Grifith	TBS	4:05:00pm	4:35:00pm
6	Content Record	NBA Basketball	TBS	4:35:00pm	6:05:00pm

Broadcast Advertising Data

		Content ID	Channel ID	Start Time	End Time
1	Content Record	Coca Cola #10	NBC	4:00:30pm	4:01:00pm
2	Content Record	Visa #2	NBC	4:01:00pm	4:01:30pm
3	Content Record	Delta #1	TBS	4:03:30pm	4:04:00pm
4	Content Record	Delta #3	TBS	4:04:00pm	4:04:30pm
5	Content Record	Visa #21	TBS	4:04:30pm	4:05:00pm

Parse and Merge Data

* → Also would include BellSouth AD insertions

Combined Data

	Subscriber	Timestamp	App. ID	# Bytes to Follow	Application Specific Data Event ID	Channel ID	Content ID
1	Event Record	3:59:30pm	Cable App.	6 bytes	Soft Power On	ABC	National News
2	Event Record	4:00:00pm	Cable App.	6 bytes	Channel Up	NBC	Murphy Brown
* 3	Event Record	4:00:30pm	Cable App.	6 bytes	Change Content	NBC	Coca Cola #10
* 4	Event Record	4:01:00pm	Cable App.	6 bytes	Change Content	NBC	Visa #2
* 5	Event Record	4:01:30pm	Cable App.	6 bytes	Change Content	NBC	Murphy Brown
6	Event Record	4:03:17pm	Cable App.	6 bytes	Channel Up	TBS	N.G.Explorer
* 7	Event Record	4:03:30pm	Cable App.	6 bytes	Change Content	TBS	Delta #1
* 8	Event Record	4:04:00pm	Cable App.	6 bytes	Change Content	TBS	Delta #3
* 9	Event Record	4:04:30pm	Cable App.	6 bytes	Change Content	TBS	Visa #21
* 10	Event Record	4:05:00pm	Cable App.	6 bytes	Change Content	TBS	N.G.Explorer
* 11	Event Record	4:05:00pm	Cable App.	6 bytes	Change Content	TBS	Andy Grifith
12	Event Record	4:06:25pm	Cable App.	6 bytes	Channel Dwn	NBC	Murphy Brown
13	Event Record	4:15:45pm	Cable App.	6 bytes	Channel Up	TBS	Andy Grifith
* 14	Event Record	4:35:00pm	Cable App.	6 bytes	Change Content	TBS	NBA Basketball
15	Event Record	4:55:45pm	Cable App.	6 bytes	Soft Power Off	None	

Start Time End Time

Figure 26: Example Content Merge

7. Clickstream Sizing Estimates

Two sets of numbers were put together to start to get a handle on possible data rates and storage needs for the network, and the back end systems to support the Clickstream system.

7.1. Maximum Sizing Estimates

This is a maximum possible approach to understand the absolute upper limit on clickstream bit rates, and storage capacities during the trial period.
For the first analysis every STB in the system is being used 24hours a day, 7 days a week, and someone is changing broadcast channels once a second. This is an extreme scenario, but one which should present us with absolute maximum data and storage rates for the Clickstream system.

Assumptions:
- Maximum possible click rate after STB filtering = **1 click per second** uniformly distributed over 24hr x 7days .
- 4000 subsribers for the trial period.
- Uniform upload process through appropriate control mechanisms.
- Average number of bytes per click = 14 uncompressed.
- Compression rate of 50% can be acheived for uploads.
- 70 Broadcast channels in system

Conclusions:
- Max number bits/second uploaded through system:
  224,000 bits/second for system
  7,000 bits/second per RT
  28,000 bits/second per Slotted-Aloha Modulator
  (18% of gross bandwidth, about 36 % of net bandwidth)
- Max number bytes of raw data to be stored per day:
  - Clickstream: **4.838 Gigabytes per day**
  - Content Metadata: **2 Megabytes per day**

7.2. Reasonable Sizing Estimates

For the second analysis, every STB in the system is being used 24 hours a day, 7 days a week, and someone is changing the broadcast channel once a minute. This is a more reasonable scenario in that during peak "channel Surfing" data rates are likely to be higher on a per STB basis. This is balanced with the fact that neither duration of viewing or overall system viewship rates are expected to every be this high.

Assumptions:
- Reasonable click rate after STB filtering = **1 click per minute** uniformly distributed over 24hr x 7days .
- 4000 subsribers for the trial period.
- Uniform upload process through appropriate control mechanisms.
- Average number of bytes per click = 14 uncompressed.
- Compression rate of 50% can be acheived for uploads.
- 70 Broadcast channels in system

Conclusions:
- Max number bits/second uploaded through system:
  3,733 bits/second for system
  117 bits/second per RT
  467 bits/second per Slotted-Aloha Modulator
  (.03% of gross bandwidth, about .06% of net bandwidth)
- Max number bytes of raw data to be stored per day:
  - Clickstream: **80 Megabytes per day**
  - Content Metadata: 2 Megabytes per day

8. MKIS System

8.1. Persistent Storage

8.2. Hardware Interfaces

8.2.1. Staging Server

8.2.2. Cable Data

8.2.3. IMS (indirect?)

8.2.4. I3/Optimark Upload

8.2.5. 3rd Party Analysis Upload

8.2.6. Marketing Report Generation Ethernet Interface

8.3. Content Merge Engine

8.3.1. Prevue Metadata

8.3.2. Advertisement Metadata

8.3.3. Advertising Traffic Controller Metadata

8.3.4. IMS Generated Metadata

8.4. Analysis Capabilities

8.4.1. Rough Overview

9. Defining Terms

The following are some terms used in this document and alternatives you may have encountered elsewhere:

I3: Formerly CONDO : Consortium of New Database Operators. A group of companies which will be providing an Analysis capability for Clickstream information, and will act as an interface between the advertising community and BellSouth.

Clickstream System: A demographics/programming ratings collection system which will function on the BellSouth IVSN. This document is an initial attempt to define the scope of this functionality, and the means we may use to make it happen.

STB: Set-Top Box. The "Cable Converter" that sits on "Top" (in this case possibly on bottom) of the subscribers television set. The device provides a platform on which content is converted to the NTSC video format and presented to the subscriber, and input is taken from the subscriber and transmitted to the cable operator. Analogous Terms: Set-Top Terminal (STT), Cable Converter, Home Communications Terminal (HCT), "Richmond"(Scientific-Atlanta model name for this device), 8600X, 8600XDi.

Subscriber: The end user of the system, the person who subscribes to the service. Traditionally in consumer electronics the "consumer" is the end user of the products or systems produced. In Cable Television the "consumer" is typically a cable operator or large Multiple System Operator (MSO), so the term subscriber was adopted.

Upstream: Generic term used for a data path provided by any mechanism FROM the subscriber to the headend, server, or billing and management databases. If the system is viewed as billing and management databases at the top, video server and headend below, Level 1 system below that, and finally the subscriber and the STT at the bottom, then data flows "Upstream" or "Downstream". Analogous Terms: Reverse data path.

TDMA, slotted ALOHA: The two mechanisms for upstream communication from the STT to the server.

Downstream: Generic term used for a data path provided by any mechanism FROM billing or management computer to server or headend, and FROM server/headend to the subscriber/STB. Analogous Terms: Forward data path.

Event: An action or change in a STT state deemed important to the building of a knowledge base on the subscriber. Examples include key presses to change channels, enter Navigator, turn STT off/on, etc. Also includes passive changes such a change of programming content while STT tunes single channel such commercial break, changes to different commercials, changes to next program, etc.

Clickstream: Term used to describe a flow of STT event data upstream to the HP server complex. The Clickstream then flows further upstream to the I3 database where demographic and programming ratings intelligence can be gathered and formatted.

ATOM: A content identifier coding scheme. ATOM codes will be used to determine programming ratings and subscriber viewing habits.

PowerTV: The operating system used on this model of STB. It must provide for most, if not all of applications functionality through the use of APIs.

John Batten

7/8/96

Date

Ted Grauch

7/8/96

Date

Tom Danner

7/8/96

Date

Witnessed:

Signature

JILL NORRIS

Typed or written name

7/8/96

Date

Witnessed:

Signature

JAMES H. GRAY

Typed or written name

7/8/96

Date